Master's thesis
Planetary geophysics

Identification of asteroid streaks in simulated ESA Euclid images

Mikko Pöntinen

May 16, 2018

Tutor: Mikael Granvik

Censors: Ilmo Kukkonen
Mikael Granvik

UNIVERSITY OF HELSINKI
Department of Physics

P.O. Box 64 (Gustaf Hällströmin katu 2a)
FI-00014 University of Helsinki

“The dinosaurs became extinct because they didn’t have a space program.”

—Larry Niven

HELSINGIN YLIOPISTO — HELSINGFORS UNIVERSITET — UNIVERSITY OF HELSINKI

Tiedekunta — Fakultet — Faculty	Laitos — Institution — Department
Faculty of Science	Department of Physics

Tekijä — Författare — Author
Mikko Pöntinen

Työn nimi — Arbetets titel — Title
Identification of asteroid streaks in simulated ESA Euclid images

Oppiaine — Läroämne — Subject
Planetary geophysics

Työn laji — Arbetets art — Level	Aika — Datum — Month and year	Sivumäärä — Sidoantal — Number of pages
Master's thesis	May 16, 2018	91 pages

Tiivistelmä — Referat — Abstract

One of the main factors currently limiting geophysical and geological studies of asteroids is the lack of visual and near-infrared (Vis-NIR) spectra. European Space Agency's upcoming Euclid mission will observe up to 150,000 asteroids and gather a large amount of spectral data of them in the Vis-NIR wavelength range. Asteroids will appear as faint streaks in the images. In order to exploit the spectra, the asteroids have to first be found in the massive amounts of data to be obtained by Euclid.

In this work we tested two methods for detecting asteroid streaks in simulated Euclid images. The first method is StreakDet, a software originally developed to detect streaks caused by space debris. We optimized the parameters of StreakDet, and developed a comprehensive analysis software that can visualize and give statistics of the StreakDet results. StreakDet was tested by feeding 4096×4136 pixel images to the software, which then returned the coordinates of the asteroids found.

The second method is machine learning. We programmed a deep neural network, which was then trained to distinguish between asteroid images and non-asteroid images. Smaller images were used for this binary classification task, but we also developed a sliding window method for analyzing larger images with the neural network.

After optimizing the program parameters, StreakDet was able to detect approximately 60% of asteroids with apparent magnitude $V < 22.5$. StreakDet worked better for long streaks, up to 125 pixels (corresponding to an asteroid with a sky motion of 80 "/h) while streaks shorter than 15 pixels (10 "/h) were typically not found. The neural network was able to classify the brightest ($20 < V < 21$) streaks with up to 98% accuracy when using very small images. When analyzing larger images, the sliding window algorithm produced heat maps as output, from which the asteroids could easily be spotted. The machine learning algorithm utilized was fairly simple, so even better results may be obtained with more advanced algorithms.

Avainsanat — Nyckelord — Keywords
Asteroids, ESA Euclid, streak detection, StreakDet, deep learning

Säilytyspaikka — Förvaringsställe — Where deposited

Muita tietoja — övriga uppgifter — Additional information

HELSINGIN YLIOPISTO — HELSINGFORS UNIVERSITET — UNIVERSITY OF HELSINKI

Tiedekunta — Fakultet — Faculty	Laitos — Institution — Department
Matemaattis-luonnontieteellinen	Fysiikan laitos

Tekijä — Författare — Author
Mikko Pöntinen

Työn nimi — Arbetets titel — Title
Identification of asteroid streaks in simulated ESA Euclid images

Oppiaine — Läroämne — Subject
Planetaarinen geofysiikka

Työn laji — Arbetets art — Level	Aika — Datum — Month and year	Sivumäärä — Sidoantal — Number of pages
Pro gradu	May 16, 2018	91 sivua

Tiivistelmä — Referat — Abstract

Yksi merkittävä asteroidien geofysikaalista ja geologista tutkimusta rajoittava tekijä on näkyvän valon ja lähi-infrapuna-alueen (Vis-NIR) spektrien puute. Euroopan Avaruusjärjestön valmisteilla oleva Euclid-avaruusteleskooppi tulee havaitsemaan arviolta 150 000 asteroidia, ja kerää niistä paljon Vis-NIR-spektridataa. Asteroidit näkyvät kuvissa himmeinä viiruina tähtitaivasta vasten. Jotta spektrejä päästään hyödyntämään, täytyy asteroidit ensin löytää suuresta määrästä Euclidin kuvadataa.

Tässä työssä tutkimme kahden menetelmän käyttöä asteroidiviirujen löytämiseksi simuloiduista Euclidin havaintokuvista. Ensimmäinen menetelmä on StreakDet-ohjelmisto, jonka alkuperäinen käyttötarkoitus on avaruusromun synnyttämien viirujen löytäminen. StreakDetiä koskeva tutkimustyö keskittyi ohjelman parametrien optimointiin ja erillisen analyysiohjelman kehittämiseen, jolla StreakDetin tuloksia voidaan visualisoida ja analysoida. StreakDetiä testattiin syöttämällä sille 4096×4136 pikselin kuvia, ja ohjelma palautti tuloksena löytämiensä asteroidien koordinaatit.

Toinen menetelmä on koneoppiminen. Ohjelmoimme syvän neuroverkon, joka opetteli opetusdatan avulla erottamaan asteroideja sisältävät kuvat kuvista, joissa asteroideja ei ole. Luokittelutehtävässä käytettiin pieniä kuvia, mutta kehitimme myös liukuva ikkuna -menetelmän suurempien kuvien analysoimiseksi neuroverkon avulla.

Ohjelmaparametrien optimoinnin jälkeen StreakDet onnistui löytämään noin 60 prosenttia näennäisen magnitudin $V < 22,5$ asteroideista. StreakDet löysi paremmin pitkiä, 125 pikselin pituisiksi yltäviä viiruja, kun taas lyhyet, alle 15 pikselin pituiset viirut jäivät yleensä löytymättä. Euclidin tapauksessa 15 pikselin viiru vastaa 10 "/h kulmanopeutta, ja 125 pikselin viiru 80 "/h nopeutta. Neuroverkko oppi erottamaan kirkkaimpia asteroideja ($20 < V < 21$) sisältävät kuvat asteroideja sisältämättömistä kuvista jopa 98% tarkkuudella, kun opetus- ja testidatana käytettiin hyvin pieniä kuvia. Suurempia kuvia analysoitaessa liukuva ikkuna -algoritmi tuotti ulostulona lämpökarttoja, joista asteroidit oli helppo havaita. Käytössä ollut koneoppimisalgoritmi oli vielä varsin yksinkertainen, joten vielä parempia tuloksia voi olla saavutettavissa kehittyneemmillä algoritmeilla.

Avainsanat — Nyckelord — Keywords
Asteroidit, ESA Euclid, viiruntunnistus, StreakDet, koneoppiminen

Säilytyspaikka — Förvaringsställe — Where deposited

Muita tietoja — övriga uppgifter — Additional information

Contents

1. Introduction

This work presents the results for two groups of methods for detecting asteroids in imaging data that simulates that to be obtained by the European Space Agency's (ESA) upcoming Euclid mission. The asteroids appear as streaks in the data, and thus the main problem to be solved is streak detection. The problem is made harder by the fact that the images contain a lot of other visual objects, such as stars and galaxies, and that not all streaks in the images all created by asteroids, but by cosmic rays instead.

Euclid is mainly a cosmological mission, focused on measuring the red shifts of galaxies, in order to shed light on the nature of dark energy. It will observe a large portion of the sky, and therefore, as a side effect, a lot of Solar System objects will appear in the data. Most of the Solar System objects will be asteroids, and as the telescope of Euclid is stabilized relative to the galaxies, the asteroids will move relative to the background sky, and show up as streaks of various lengths in the images (Carry, 2018).

In general, the importance of studying asteroids lies in three main categories. First, there are major scientific reasons for their study, namely that asteroids are key to understanding the origin and evolution of the Solar System, and that the study of numerous organic molecules found on asteroids could shed light on the origins of life on Earth. Second, asteroid impacts are an existential risk, i.e., an event that could, in the worst case, cause the extinction of humanity. Asteroid impact avoidance is of

paramount importance, and therefore it is necessary to be aware and know the orbits of all asteroids that could pose an impact threat. Third, asteroids offer immense economic potential due to the gigantic amounts of valuable materials in them. This potential can be realized through asteroid mining.

The main motivation behind this thesis is a somewhat lengthy chain of reasoning, which goes as follows. All of the aforementioned goals for studying asteroids rely on the discovery and geophysical understanding of the asteroids. Measuring and knowing the spectra of the asteroids is essential in their geophysical and compositional modeling. Currently, only a tiny fraction of the discovered asteroids have measured spectra. The Euclid mission will massively increase the number of measured asteroid spectra extending to near-infrared. The first step in the analysis of Euclid data is to find asteroids in Euclid images. This work aims to help finding the asteroids, which is an essential step before anything else in this motivational chain is possible.

The first method used to tackle the problem of finding the asteroids is a software called StreakDet, which was developed by Virtanen et al. (2016) to detect streaks caused by space debris in astronomical images. The work related to StreakDet was to systematically test its ability to find asteroids in simulated Euclid images. We did this by optimizing the parameters of StreakDet, and by developing a separate test and analysis software to give statistics on the results.

The second method used is machine learning, namely artificial neural networks. Machine learning with neural networks, especially deep ones with numerous neurons and layers, has become known as deep learning, and the method has reached remarkable milestones in the past few years in many areas such as image recognition, speech recognition, language translation and beating human world champions in complicated games such as Go (LeCun et al., 2015). In this work, the main goal related to deep learning was to carry out simple proof-of-concept tests to see whether the

algorithms could be trained to distinguish between asteroids and non-asteroids. We programmed the machine learning algorithms from the ground up, tested them first in a simple binary classification task, and then developed a preliminary algorithm to spot the asteroids from larger images.

The structure of this thesis is the following. Chapter 2 goes through the basics of asteroids, and elaborates on the categories of reasons why it is important to study them. Chapter 3 contains the basic facts about ESA's Euclid mission, and explains how the simulated data used in this work is generated. Chapter 4 explains the algorithmic pipeline of the StreakDet streak detection software, as well as the inner workings of the Python analysis software that we developed to visualize and analyze the StreakDet results. Chapter 5 explains the basics of machine learning, especially those of logistic regression and artificial neural networks, and shows the implementation on the exact machine learning algorithms and the generation of the training data used in this work. Chapter 6 contains the results for StreakDet, when it was tested on the simulated Euclid data, as well as the results for the machine learning approaches used for similar data. Chapter 7 contains discussion about the methods and results, while Chapter 8 presents the conclusions of this thesis.

2. Asteroids

Asteroids are small Solar System bodies in the inner Solar system. The significance of studying asteroids lies in three main categories:

1. Their scientific importance, mainly in understanding the origin and formation of the Solar System,
2. asteroid impact avoidance,
3. economic potential, namely asteroid mining.

2.1 What Are Asteroids

Asteroids are a subgroup of small Solar System bodies (SSSB). There are other SSSB groups as well, such as comets and trans-Neptunian objects. The lines between different SSSB populations are not always very well defined, but usually the term asteroid is used to refer to small, non-comet-like bodies in the inner Solar System, at Jupiter's orbit or closer. The main asteroid populations are near-Earth asteroids (NEA), main-belt asteroids (MBA) and Jovian Trojans. Very minor populations are Trojans of Earth, Mars, and Uranus.

NEAs have perihelion distances, the point on the orbit closest to the Sun, of less than 1.3 au [1]. An umbrella term near-Earth object (NEO) is used, if near-Earth comets (NEC) are included. Most of NEOs are NEAs, and they are divided

[1] https://cneos.jpl.nasa.gov/about/neo_groups.html

further into groups called Atira, Aten, Apollo, and Amor, determined by their orbital characteristics. Atiras have orbits that are completely within the orbit of the Earth, i.e., the aphelion of an Atira is smaller than the perihelion of the Earth. More technically, for Atiras, a < 1.0 au and Q < 0.983 au. Atens are Earth-crossing asteroids, so their perihelions are inside the Earth's orbit and aphelions are outside of it. Atens have a semi-major axis of less than 1.0 au, so for them, a < 1.0 au and Q > 0.983 au. Apollos are also Earth-crossing asteroids, and the difference to Atens is that the semi-major axis of Apollos is larger than 1.0 au, so that a > 1.0 au and q < 1.017 au. Amors are near-Earth asteroids that orbit entirely outside the orbit of the Earth.

Trojan is a term used to describe an asteroid sharing an orbit with a planet, typically orbiting the Sun at or close to the Lagrange points 4 or 5, leading or trailing the planet by 60 degrees. Earth has only one known Trojan, whereas Mars has nine known Trojans[2]. Jupiter has over 7,000 known Trojans, and is thought to have about a million more with diameters larger than 1 km, and countless smaller ones. The Jovian Trojan population could be as large as the main asteroid belt population. Uranus has one and Neptune has 17 known Trojans.

Small Solar System bodies farther from the Sun are Centaurs, Neptunian Trojans and Trans-Neptunian objects. The farthest SSSB population is the Oort cloud, which is thought to be the source of most long-period comets. The difference between asteroids and comets is mainly their composition. Asteroids are more stony, carbonic or metallic, whereas comets are mostly composed of ice and dust. Due to their composition, when comets arrive closer to the Sun, the volatiles close to their surfaces start to sublimate, and they start outgassing, and thus develop a coma and sometimes a tail. However, the distinction between an asteroid and a comet has become less clear with the discovery of so-called active asteroids, which are objects

[2]https://minorplanetcenter.net/iau/lists/Trojans.html

orbiting within the main belt, yet exhibiting comet-like behavior during some parts of their orbits (Jewitt et al., 2015).

Asteroids are divided into different spectral types, suggestive of composition, and to orbital classes by their dynamical characteristics. The main spectral complexes are C, S and X. Roughly speaking, C refers to carbon-rich, S to silicate/stony, and X to other compositions, such as metallic. With orbital classification, there are groups, which are more loose sets of asteroids lumped together according to similar orbits. Asteroid families are more tightly related, and they are formed by a break-up of a parent asteroid.

There are several slightly different spectral classification systems, the most popular ones being Tholen with 14 asteroid categories (Tholen, 1984), SMASS (2002 Small Main-Belt Asteroid Spectroscopic Survey), also known as Bus or Bus-Binzel (Bus, 1999) and Bus–DeMeo (DeMeo et al., 2009). The SMASS system is built on the Tholen system, and further the Bus-DeMeo system is built on the SMASS system, extending from the visual wavelengths into near-infrared.

Asteroids come in a large variety of sizes, ranging from 1 meter to several hundreds of kilometers. Asteroids smaller than 1 m are usually referred to as meteoroids. The largest object in the asteroid belt is (1) Ceres, with a mean diameter of 953 km, and it is thought to contain approximately one third of all the mass in the asteroid belt. Ceres was the first object to be discovered in the asteroid belt, and when Giuseppe Piazzi found it on 1 January 1801, it was originally considered a planet. A few more discoveries, (2) Pallas, (3) Juno and (4) Vesta were also classified as planets at first. In the 1850s they were re-classified as asteroids, as more objects in the main belt started to be discovered. In 2006 Ceres was again re-classified, now as a dwarf planet, although it is still often considered as also being the largest asteroid. Vesta is the second largest asteroid with a mean diameter of 525 km, closely followed by Pallas with a mean diameter of 512 km.

Historically, asteroids were often found with a blink comparator, which rapidly switches between two photographs taken from the same part of the sky. In the process, stars stay in the same place in both photographs, but asteroids and planets move, and thus finding a moving dot suggests a Solar System object in the images. For example, Clyde Tombaugh found Pluto in 1930 by using this method. Yrjö Väisälä developed a more efficient double-exposure method for finding asteroids, which works by exposing the same photographic plate two times, with a pause between the exposures and a slight offset in the positions of the stars in the photograph (see e.g., Väisälä (1939)). This way, stars appear as two dots side by side, whereas any single dot, or two dots whose distance from each other differs from that of the stars, is indicative of a Solar System object. The Minor Planet Center credits Väisälä with discoveries of 128 numbered asteroids [3]. In modern times, the images are in digital format and the asteroids are typically found with the help of computer algorithms.

A number of spacecraft have executed flybys or orbital insertions to asteroids. Flyby missions include the Galileo missions by NASA, which imaged (951) Gaspra in 1991, and (243) Ida and its moon Dactyl in 1993, during its route towards Jupiter. The first pure asteroid mission was NASA's NEAR Shoemaker, which flew by (253) Mathilde in 1997, entered into orbit around (433) Eros, and eventually landed on its surface in 2001. Other flyby missions, also by NASA, were Deep Space 1 in 1999, with an encounter with (9969) Braille before a flyby of comet 19P/Borrelly, and Stardust's flyby of (5535) Annefrank in 2002 on its way to collect and return samples from comet Wild 2. Another sample return mission was Japanese Hayabusa mission to asteroid (25143) Itokawa, which managed to return a small amount of samples to Earth in 2010. ESA's Rosetta mission flew by asteroids (2867) Šteins in 2008 and (21) Lutetia in 2010, before reaching comet 67P/Churyumov–Gerasimenko

[3] https://www.minorplanetcenter.net/iau/lists/MPDiscsNum.html

in 2014. In 2012, China's lunar orbiter Chang'e 2 flew by asteroid (4179) Toutatis on an extended mission. With the aid of ion propulsion, NASA's Dawn spacecraft has reached orbits around two different asteroids, first around Vesta from July 2011 to September 2012, and secondly around Ceres since 2015. Dawn will operate at Ceres until it runs out of hydrazine fuel, after which it will be placed in a stable orbit around the dwarf planet, where it will stay indefinitely.

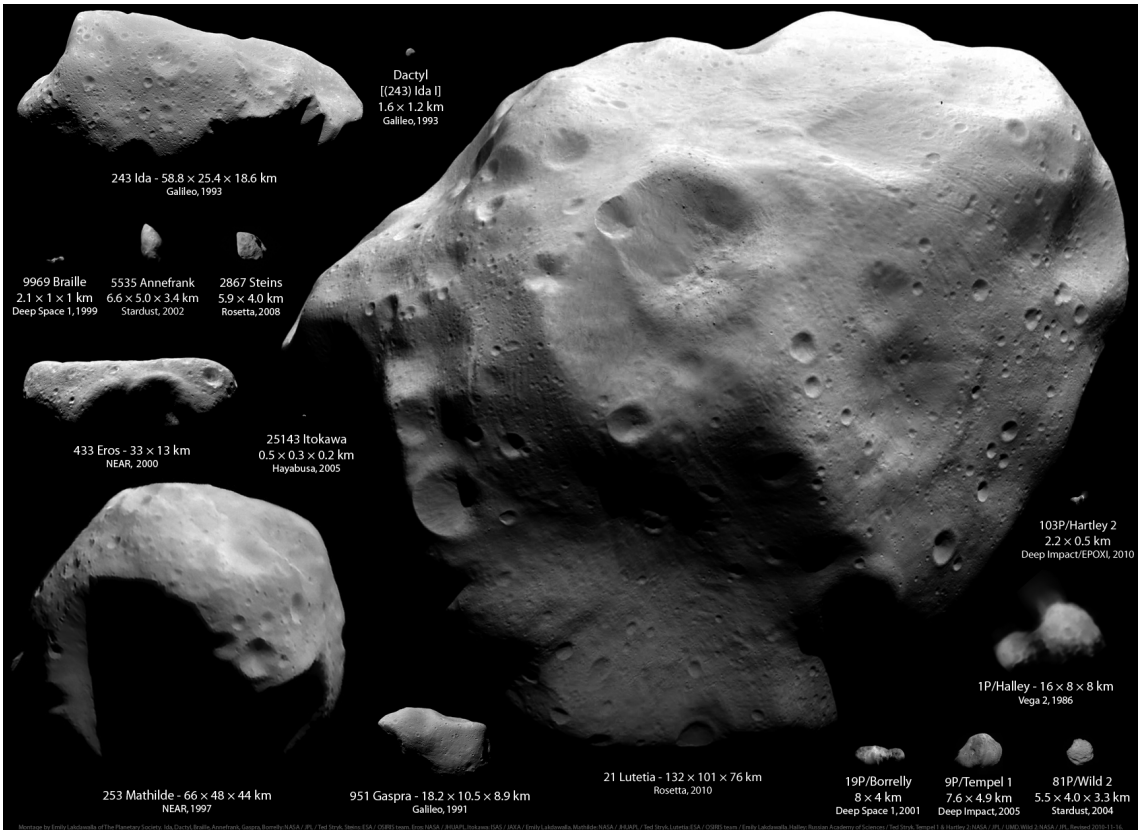

Figure 2.1: Images of most of the asteroids and comets visited by spacecrafts. (Montage by Emily Lakdawalla. Ida, Dactyl, Braille, Annefrank, Gaspra, Borrelly: NASA / JPL / Ted Stryk. Steins: ESA / OSIRIS team. Eros: NASA / JHUAPL. Itokawa: ISAS / JAXA / Emily Lakdawalla. Mathilde: NASA / JHUAPL / Ted Stryk. Lutetia: ESA / OSIRIS team / Emily Lakdawalla. Halley: Russian Academy of Sciences / Ted Stryk. Tempel 1, Hartley 2: NASA / JPL / UMD. Wild 2: NASA / JPL)

Current missions to asteroids are Japanese Hayabusa 2, which aims to return samples from asteroid (162173) Ryugu in December 2020, and has been launched

in December 2014, and NASA mission OSIRIS-REx which aims to return samples from asteroid (101955) Bennu, having been launched on September 8, 2016. Future missions include NASA missions Psyche to metallic asteroid (16) Psyche, and Lucy to visit several Jupiter Trojans. NASA is also planning an asteroid capture mission, possibly in the 2020s.

Small solar system objects are known to exist around other stars as well. They have been detected using the thermal infrared emissions of their collisional dust disks (Lawler and Gladman, 2012), and the changes of the spectra of cool white dwarf stars caused by comets (Alcock et al., 1986) and asteroids (Jura, 2008) falling into them.

On 19 October 2017, the first interstellar asteroid, (1I/2017 U1) 'Oumuamua, was observed directly, as it was flying through the Solar System in a hyperbolic orbit. 'Oumuamua appears to be highly elongated, with a length of approximately 10 times its width. Possible dimensions could be 800m x 80m x 80m (Meech et al., 2017).

2.2 Scientific Significance of Asteroids

The scientific importance of asteroids ultimately stems from two major goals. The first goal is understanding the origin and formation of the Solar System, and the second goal is understanding the origin of life. The reason for asteroids being valuable for the study of the origin of the Solar System is that asteroids contain primordial material that is hardly available elsewhere, except on comets. The fact that asteroids can be utilized in the study of the origin of life is explained by the fact that many organic compounds, including relatively complex amino acids, have been found in many meteorites. These amino acids and other organic molecules are the building blocks of life, so understanding which of these building blocks were available on the early Earth can help shed light on the formation of the first lifeforms.

Naturally, under the aforementioned two main point of interest there are countless numbers of open scientific sub-problems, relating to the dynamical and geophysical properties of the asteroids. One recent example is the discovery by Granvik et al. (2016), which shows that a high portion of asteroids whose perihelion is in the order of a few tens of solar radii, get destroyed in a relatively short period of time. The destruction mechanism is not currently known, as the asteroids are destroyed farther from the Sun than simple evaporation of material would explain. Solving sub-problems like this are steps along the way towards understanding the important, large questions.

The following paragraphs describing the basic scientific approach to asteroids and their spectra are mostly based on the review by DeMeo et al. (2015).

There are currently over 700,000 asteroids for which we have at least some orbital information. About 100,000 asteroids have at least some form of measurements that provide information of their surface compositions. The most important surveys that have provided information on asteroid compositions are Sloan Digital Sky Survey (SDSS), which is primarily a cosmological survey, but offers wide-band photometry of more than 100,000 asteroids (Ivezić et al., 2001), and the Wide-field Infrared Survey Explorer (WISE), which has provided diameter and albedo estimates for over 100,000 asteroids (Mainzer et al., 2011). Gaia is currently obtaining narrow-band photometry of hundreds of thousands of asteroids but the data is yet to be published. In a few years, the Large Synoptic Survey Telescope (LSST) will start operating, providing accurate astrometry and multi-color photometry for a large number of Solar System objects (Jones et al., 2009). And, of course, Euclid will be launched in a few years.

Typical methods for studying the asteroid surface properties, such as mineralogy, grain size, and space weathering, are spectroscopic, photometric, and polarimetric observations in wavelengths ranging from ultraviolet to infrared. Surface

albedos, related to surface compositions, can be calculated by observing thermal infrared emissions of the asteroids to obtain the size of the asteroid, and compare them to observations in the visual wavelengths. Broad absorption and emission features in the asteroid spectra can provide information about the minerals, but a relatively small number of high-quality spectra have been obtained so far. A further challenge is the determination of surface composition from spectra, because many surface materials do not produce clear absorption features, and because the spectrum is also affected by grain size, space weathering, temperature, and viewing geometry. For taxonomic classification, spectra with less resolution is adequate, which is also the reason for there existing taxonomic classifications for several orders of magnitude more asteroids than for which there exists high-quality spectra.

The three major taxonomic groups, S, C and X have distinct spectral features, and are divided further into classes or types. S-complex has moderate absorption features at 1 and 2 microns, suggestive of silicate composition. The C-complex has been named after carbonaceous chondrite meteorites, and C asteroids have low albedos with relatively flat spectral slopes. They have few absorption features, with an exception being a 0.7-micron feature suggestive of phyllosilicates. The X-complex spectra have moderate slopes and subtle or no features. The X-complex consists of many different types of asteroids, with albedos ranging from a few percent to more than 50 percent. In the Tholen system, the X-complex is divided further into E, M, and P classes, depending on albedo, while in the Bus-Demeo system the classes are X, X_c, X_e, and X_k. Some classes that do not fit in any of the three main complexes are D-types with very red slopes, A-types with spectral features in 1 micron region related to olivine, V-types with spectral features suggestive of pyroxene, and K, L, T, O, Q, and R-types.

The different asteroid types are not uniformly distributed in the main asteroid belt. The Hungaria region ($1.8 < a < 2.0$ au) consists mostly of E-type asteroids

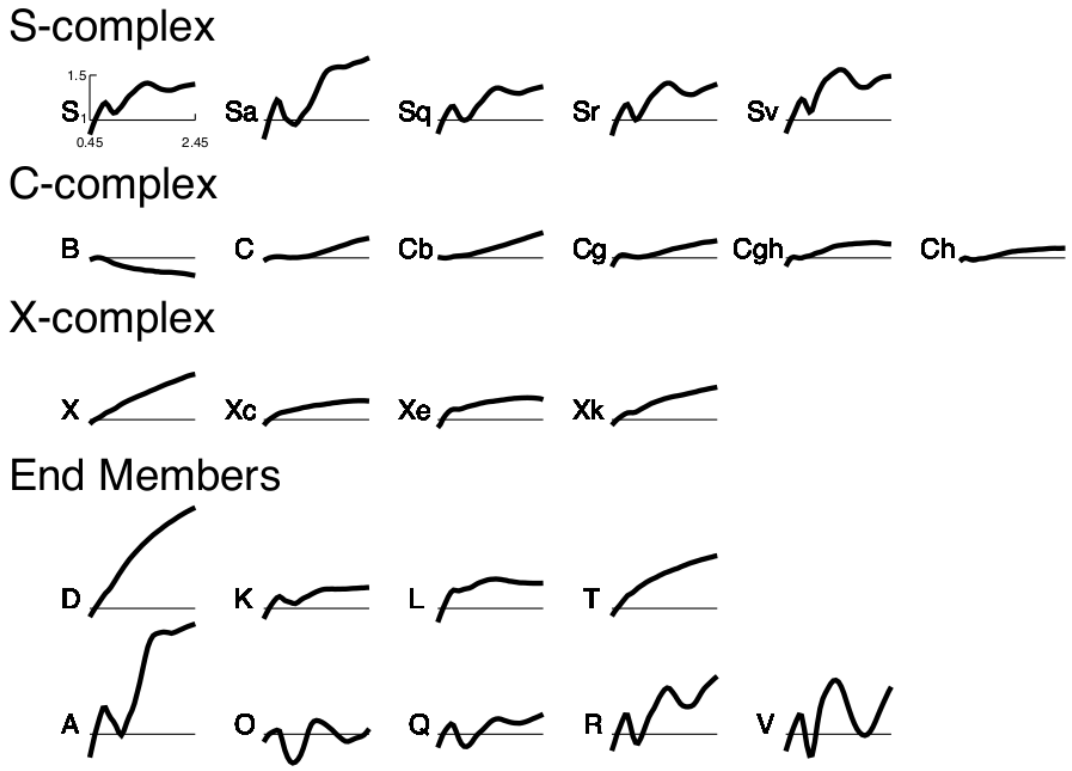

Figure 2.2: The Bus-Demeo taxonomic classification system with 24 spectral classes. The spectra are in the visible and near-infrared regions. X-axes mark the wavelength, ranging from 0.45 and 2.45 microns, and y-axes mark the relative reflectance, ranging from 1 to 1.5 above the x-axes. The figure is from DeMeo et al. (2015).

with albedos of more than 0.3 and typically with spectral features at 0.49 microns, and they are part of the Hungaria asteroid family, broken off from asteroid (434) Hungaria. In addition to E, the Hungaria region also includes S-types and C-types.

The Inner Main Belt ($2.0 < a < 2.5$ au) contains asteroid (4) Vesta, and its smaller V-type family members. Although numerous, the other Vesta family asteroids are typically small and thus do not contribute a lot to the mass of the inner belt, when the actual (4) Vesta is excluded. In addition to V, there are several large S-type asteroids. There are only a few large (diameter $D > 100$ km) C-type asteroids in the Inner Belt, but in the medium ($20 < D < 100$ km) diameter range they are numerous and equal to approximately quarter of the mass, and in small

sizes ($5 < D < 20$ km) they are as numerous as S-types. There are also a number of M-types, P-types and D-types.

The Middle Main Belt ($2.5 < a < 2.82$ au) has C-type Ceres and B-type Pallas, which contain around 31% and 7% of the mass of the whole Main Belt. In the smaller size range, the Middle Belt is very similar to the Inner Belt.

The Outer Main Belt ($2.82 < a < 3.3$ au) consists predominantly of C-complex, with the lead of (10) Hygiea. Although the percentage of S-complex is relatively small, their total mass is quite high, because the Outer Belt is several times more massive than the Inner Belt.

The largest Hildas ($a \sim 4$ au) are mainly P-type and the largest Jupiter Trojans ($a \sim 5.2$ au) are mostly D-type.

When the Solar System was formed by condensing from a disk of uniform density, there should have been around one Earth-mass worth of material in the asteroid belt (Weidenschilling, 1977). In reality, the total mass of the Main Asteroid Belt is only approximately 5×10^{-4} Earth masses (Krasinsky et al., 2002). Similarly, the mass of Mars is too small. The Grand Tack Model tries to explain these discrepancies by suggesting that the gas giants Jupiter and Saturn were migrating in the early Solar System, and as Jupiter first migrated inwards, it depleted the asteroid belt, and as it migrated back outwards, it scattered some of the asteroids back to the main belt, and in the furthest part it caused some populations of outer solar system objects to move to the asteroid belt (Walsh et al., 2011). A related hypothesis is the Nice Model, which suggests another migration of giant planets roughly 400 million years later (Tsiganis et al., 2005; Morbidelli et al., 2005; Gomes et al., 2005), and which could be responsible of altering the orbits of the asteroids, and further depleting the population (Morbidelli et al., 2010).

The number of asteroid samples retrieved from sample return missions is extremely limited, but there is a much larger number of naturally delivered asteroid

samples, namely the meteorites. They are almost always parts of asteroids that have survived all the way to the surface of the Earth, although there are also some meteorites from the Moon and Mars. Different types of meteorites have been studied a lot in the laboratories and their properties are well-known. One of the current major research topics is linking the meteorites to their parent bodies. In general, the linking is very difficult, except for the lunar and Martian meteorites.

Meteorites are separated into two main groups, chondrites (unmelted) and non-chondrites (melted). As the asteroids formed, they were mostly heated by radioactive aluminium-26. The parent asteroids of chondrites were probably smaller asteroids, which had lower inner temperatures, or were formed later, when aluminium-26 was more depleted, and thus avoided melting and were not differentiated. Some of the chondrites may originate from the unmelted crusts of otherwise internally differentiated parent asteroids. Chondrites are divided into enstatite chondrites, ordinary chondrites, carbonaceous chondrites, and rarer Kakangari and Rumuruti types.

The chondrites contain chondrules, which are enclosed in the surrounding material, called matrix. The carbonaceous chondrites contain refractory inclusions, which consist of calcium-aluminium-rich inclusions (CAIs) and amoeboid olivine aggregates (AOAs). CAIs are the oldest solid materials in the Solar System, with ages dated to 4567.3±0.2 Myr (Connelly et al., 2012). The formation of chondrules seems to have occurred 1–3 million years after the refractory inclusions, as molten silicate-metal droplets solidified, although some of them started forming already at the same time as the CAIs. The matrix consists of fine-grained crystalline and amorphous silicates, but has often been modified in the parent body. The matrix also contains volatile elements and organic matter.

The non-chondrites are divided further into primitive achondrites, which have been through extensive metamorphism and some partial melting, and into differentiated meteorites, which have experienced more comprehensive melting and dif-

ferentiation. The differentiated meteorites can be divided into achondrites (stony), stony-irons and irons. They represent the part of which the meteorite originates from the differentiated parent body, i.e., many achondrites come from the crust of differentiated bodies, while stony-irons, such as pallasites, come possibly from core-mantle boundary, and many irons come from the core.

Spectral analysis methods of asteroids have advanced in recent years. Mainly, there are now better models for reversing the effects caused by temperature, phase angle (i.e., the Sun-Asteroid-Observer angle), and grain size in the spectra, especially for S-type and V-type asteroids (Reddy et al., 2015). Different temperatures of asteroid surfaces cause changes in the properties of absorption bands, such as their depth, width, center location, and area ratio. For main belt asteroids, the phase angle is typically less than 25°, but for near-Earth asteroids it can be much larger. When the phase angle increases, the spectrum appears redder due to the fact that single scattering albedo has a wavelength dependence. This effect is known as phase reddening. The phase angle also affects albedo and depths of absorption bands. In turn, grain size affects the overall reflectance of an asteroid, and the slope and depth of the absorption features in its spectrum. For determining asteroid mineralogies, there are helpful formulas for a crude analysis of spectra that contains clear absorption bands at 1 and 2 microns. Typically, the minerals in asteroids are olivine, which has absorption features at 1 micron, and pyroxene which has absorption bands at both 1 and 2 microns. Farther in the infrared region, hydrated silicates cause absorption features in the 3-micron region. Modified Gaussian models (Sunshine et al., 1990) can be used to break down the absorption features of a spectrum, in order to determine the individual minerals contributing to the features. If no clear absorption bands are observable in the spectra, radiative transfer models can be used.

2.3 Asteroid Impacts

Since the discovery of ancient asteroid impacts on Earth, one of the main motivations of asteroid research has become the discovery and mapping of potentially hazardous asteroids (PHA) among the near-Earth asteroid population. A known example of a baneful asteroid impact is the Chicxulub impact, created by an asteroid or comet between 10 to 15 km in diameter. The impact caused the mass extinction of dinosaurs around 66 million years ago in the Cretaceous–Paleogene (K–Pg), aka Cretaceous–Tertiary (K–T) extinction event (Alvarez et al., 1980; Hildebrand et al., 1991). The most well-known asteroid impact in the relatively recent history is the the Tunguska event, which was a huge explosion in Siberia, Russia on 30 June 1908. The explosion flattened around 2000 km^2 of forest, and is thought to have been caused by an asteroid that disintegrated and exploded in the atmosphere. The most recent notable impact was that of the Chelyabinsk meteor on 15 February 2013 in Russia. The asteroid in question was approximately 20 meters in diameter, and it entered the Earth's atmosphere at around 19 km/s, before exploding in the atmosphere at an altitude of around 30 km with an energy of approximately 30 times more than the Hiroshima atomic bomb (Popova et al., 2013).

Among all the Small Solar System body populations, the most probable Earth impactors are Near-Earth Asteroids (NEA). More precisely, the potentially hazardous asteroids (PHA) are defined as NEAs, whose minimum orbit intersection distance (MOID) with respect to the Earth is less than 0.05 au and the absolute magnitude (H) is 22.0 or less. An $H = 22$ typically corresponds to a diameter in the range from 100 to 150 meters. The level of hazard imposed by a PHA is estimated with the Palermo Technical Impact Hazard Scale, which compares the hazard of an asteroid to the background hazard, i.e., the average risk imposed by other asteroids of at least the same size until the expected impact date (Chesley et al., 2002). The Palermo scale is logarithmic, with value 0 being equal to the background hazard,

value 1 indicating a hazard 10 times larger than the background hazard, value 2 marking a hazard 100 times larger than the background, and so on. Another scale, aimed at the general public, is the Torino Impact Hazard Scale, which is a simpler scale running from 0 to 10, expressing the impact probability within the next 100 years and the impact energy (Binzel, 2000).

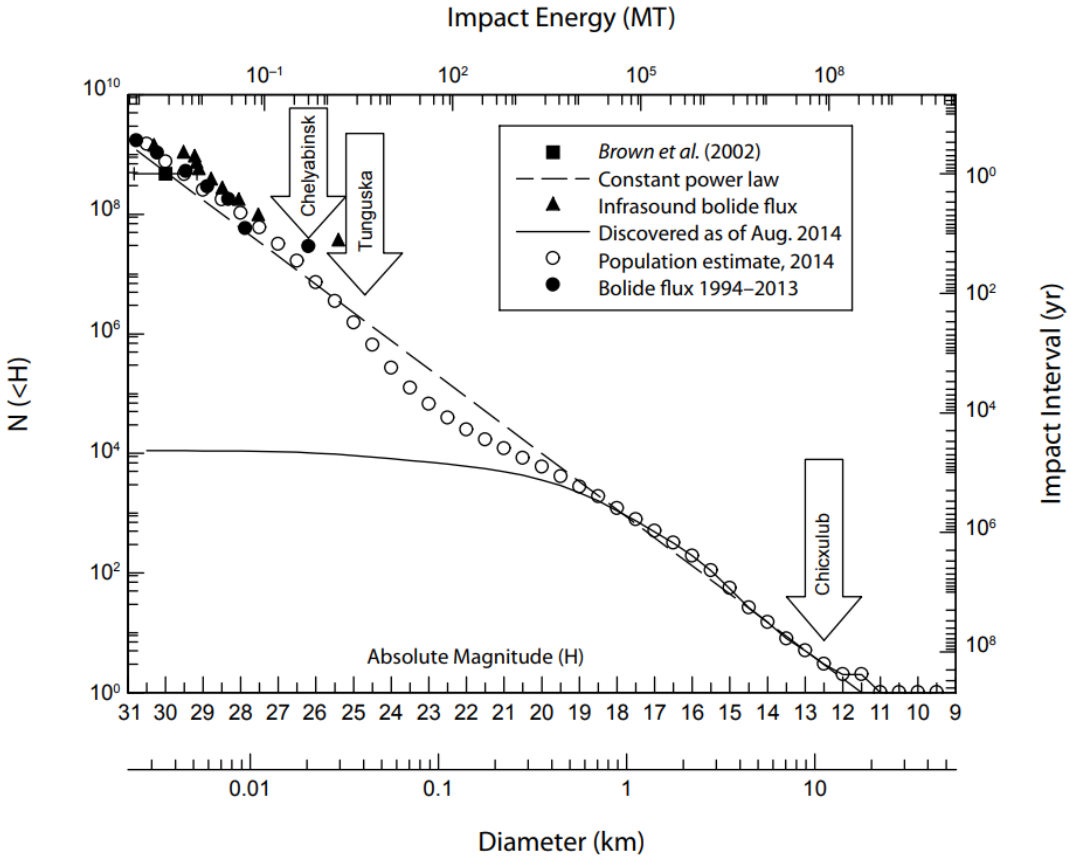

Figure 2.3: A plot of the estimated number of near-Earth asteroids and their average impact intervals to Earth, as a function to the size/magnitude of the objects. The continuous curve shows the number of known NEAs, the empty circles show the estimated total number of NEAs from models using discovery and redetection rates. The filled squares, triangles and circles represent measured impact fluxes. The main insight of the plot is that smaller objects are much more numerous, as are their impacts. Larger impacts are more rare but more devastating. The figure is from Harris et al. (2015), containing data from Brown et al. (2002).

Currently there are no confirmed NEAs that will impact the Earth in the

near future, and no objects with positive ratings in the Palermo scale, or non-zero ratings on the Torino scale [4]. If a notable impactor were to be found, and the impact date was in relatively near future, i.e., not hundreds or thousands of years away, some form of asteroid impact avoidance would have to be developed, built and used. Potential methods include things such as nuclear bombs, kinetic impactors and gravity tractors. A nuclear bomb could be used either to completely destroy the asteroid, or to evaporate material from one side of the asteroid, causing a rocket-motor-type effect to deflect the asteroid from its course. An explosion that would break the asteroid into pieces but not deflect them enough to cause them to miss the Earth, could even be counterproductive, as many small impactors could cause more damage than a single larger one. In general, its easier to slightly modify the asteroid's orbit so that it would fly past the Earth instead of impacting it. The earlier the deflection is executed, the smaller it needs to be, because even a small change in orbit accrues over time. The kinetic impactor would be a fast moving spacecraft that would impact the asteroid and thus change its orbit. A kinetic impactor prototype has been tested in NASA's Deep Impact mission, which successfully released an impactor to hit comet Tempel 1 (9P/Tempel) in 2005 with a relative velocity of around 10 km/s, and managed to slightly change the comet's orbit (A'Hearn et al., 2005). The gravity tractor would be a heavy spacecraft that would stay on one side of the asteroid, typically with the aid of ion propulsion, and this way the spacecraft would gravitationally attract the asteroid, slowly changing its orbit. There are also numerous other potential methods.

A less likely but potentially more hazardous impactor than a NEA would be a long-period comet. The reason is that the impact speed would be higher than for an NEA. NEAs are almost by definition in quite similar orbits as the Earth, so the relative velocities are not very large, but long-period comets are on very eccentric

[4]https://cneos.jpl.nasa.gov/sentry/

orbits, falling towards the Sun from the outer Solar System, resulting in a high velocity relative to the Earth. Another reason, related to impact avoidance, is that an impacting long-period comet would likely be discovered only some months before the impact, which would make impact avoidance very difficult and time-critical.

As a philosophical point, it has been argued that decreasing the likelihood of human extinction should be one of the main priorities of mankind (Bostrom, 2013). The starting point of the argument is that there could be a lot more future humans than there are currently alive. If humanity goes extinct, in addition to the deaths of all the living humans, all the possible future lives will never come into existence and thus are lost as well. If it is accepted that future human lives are as important as the lives of current humans, it follows that decreasing the existential risk even by a tiny fraction corresponds to saving a huge number of potential human lives. For example, if the Earth remains habitable for another billion years, and can sustain a billion people at a time, there could be around 10^{16} future human lives on the Earth, if every human were to live to be 100 years old. Thus, as an expected value, reducing the existential risk by just one millionth of a percentage point would correspond to saving the lives of a 100 million people. Taking an even bolder view, if humans will spread out into space, the future accessible universe could contain possibly around 10^{32} human lives. If the future minds are to be implemented as computer emulations instead of biological beings, the accessible universe offers a lower bound of 10^{52} simulated human (or other) lives of 100 years long. Especially in these latter scenarios, reducing the existential risk even with minuscule amounts gives astronomically high expected returns. Because major asteroid impacts are one of the existential risks, studying and preventing them is very important.

2.4 Economic Significance of Asteroids

The third motivation to study asteroids is their economic potential. Currently it is very expensive to launch anything from Earth to space, which severely limits constructing any kind of large scale space stations and other structures. Earth's gravity well is so deep that most of the Δv of rockets is needed to go from the surface to low Earth orbit (LEO). Once in LEO, most of the work is done, and less Δv is needed to get to basically anywhere in the Solar System. Instead of bringing materials from Earth to space, it would be economically more sensible to acquire raw materials from space, in situ, from near-Earth asteroids.

The most valuable raw material, especially in the beginning of asteroid mining business, would be water (Lewis, 2015). One of the main forms of rocket fuel, liquid oxygen and liquid hydrogen, can be produced directly from water by electrolysis. Therefore, water could be mined from near-Earth asteroids, be turned into rocket fuel and then be used to refuel satellites and spacecrafts in orbit. In a later stage, the mining of precious and non-precious metals could become economically viable. Metals such as iron could be used for construction projects in space, whereas precious metals such as gold could be brought down to Earth. On Earth, most of the heavier elements have sunk to Earth's core during the formation and differentiation of the planet. On the other hand, on asteroids many of these metals are common and readily available, especially on C-complex and M-type asteroids.

Admittedly, the engineering challenges of mining and processing valuable materials in near-zero-gravity are nontrivial. Nevertheless, encouraged by the possible economic feasibility of the enterprise, there are several companies pursuing a road to the asteroids. The largest private companies aiming for asteroid mining are Planetary Resources and Deep Space Industries. Also, the government of Luxembourg has reserved a budget of 200 million euros for the preparation of asteroid mining activities.

Potentially hazardous asteroids are often prime candidates for asteroid mining due to the fact that they are relatively similar orbits with Earth, which implies low Δv requirements and easy access. Provokingly put, there is a choice between mining the potentially hazardous asteroids for useful resources and utility, or doing nothing and waiting for some of them to eventually impact the Earth.

3. Euclid Mission

The main aims of the Euclid mission are to study the effects that dark energy has played in the expansion history of the universe, and the formation of large-scale cosmic structures. As a side effect, it will observe also up to 150,000 small Solar System objects.

3.1 Mission Profile

Euclid is classified as a medium class (M-class) astronomy and astrophysics mission, under the Cosmic Visions program. Euclid was selected for execution in October 2011. The current planned launch date, after having been postponed a few times, is in 2021 from Kourou, French Guiana with a Soyuz rocket. The planned mission duration is 6 years and 3 months, a time during which Euclid will stay in a Halo orbit in the proximity of the L2 Lagrange point of the Sun-Earth system. The spacecraft and the Service Module will be built by Thales Alenia Space, and the Payload Module will be built by Airbus. The mission is named after the famous mathematician of ancient Greece, Euclid of Alexandria.

Euclid is mainly a cosmological mission, and it will survey a large portion of the sky, approximately 15,000 deg^2 (Amendola et al., 2016). The main goal of the mission is to measure the redshifts of galaxies, in order to determine the relationship between distance and redshift. This relationship will reveal details of dark energy and how it has affected the expansion of the universe. Simply put, Euclid aims to

measure the geometry (thus the name Euclid) and the acceleration of the universe. The redshifts measured will span out to approximately 2, which corresponds to observing the distant galaxies as they appeared around 10 billion years ago. The second aim of the mission is to study the large scale structures and clusters and their formation. Gravitational lensing will be exploited in the measurements of the galaxies, as well as baryon acoustic oscillations, and spectroscopic measurements of the objects. Because gravity lensing and clustering are affected by dark matter, Euclid will help studying the properties of dark matter as well.

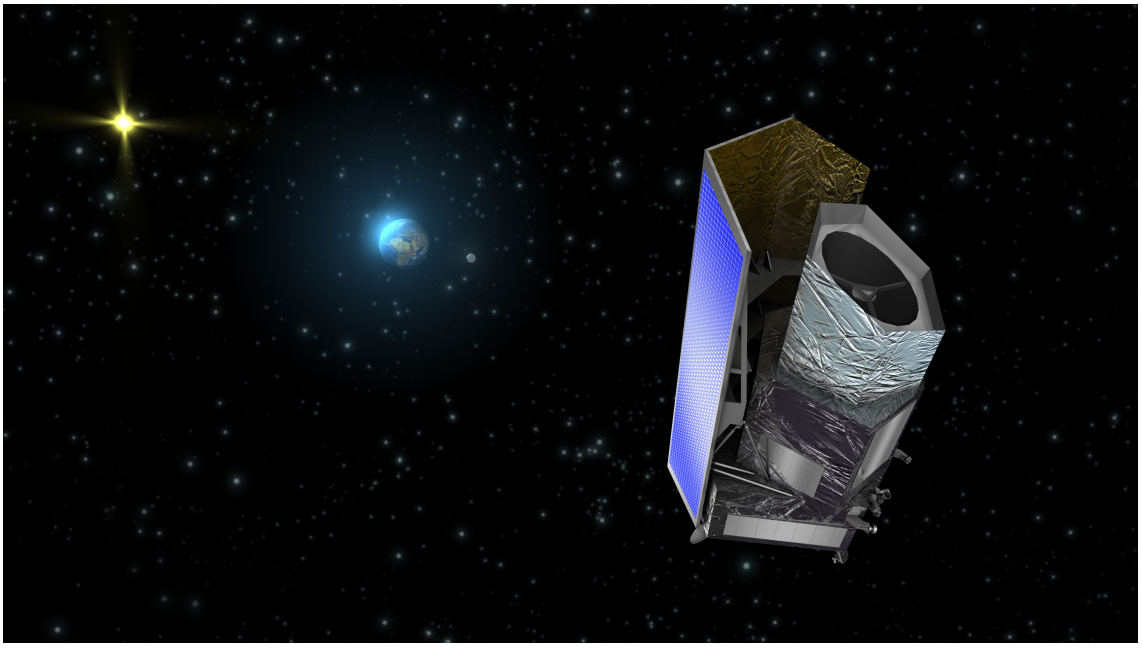

Figure 3.1: An artist's visualization of the Euclid spacecraft in operation. (ESA)

Euclid's main instrument is a Korsch-type, silicon carbide mirror telescope with a diameter of 1.2 meters and focal length of 24.5 meters (Joachimi, 2016). The telescope operates between wavelengths of approximately 550 nm (green) and 2000 nm (near-infrared). The measuring instruments are VIS (VISual imager) and NISP (Near-Infrared Spectrometer and Photometer), both of which have a 0.57 deg^2 field of view. VIS is a 500 megapixel high-quality panoramic visible imager, operating between the wavelengths of 550 and 900 nm. NISP consists of near-infrared 3-

filter photometer (NISP-P) and a slitless spectrograph (NISP-S). The near-infrared spectra will have a resolving power of 380. The pixel size for VIS is 0.1 arcseconds per pixel, and for NISP 0.3 arcseconds per pixel, corresponding to diffraction limits of 0.6 and 1.7 microns, respectively. Euclid will operate in a step-and-stare mode. Both instruments will observe an area of the sky with an exposure time of 565 seconds, after which NISP will execute three shorter exposures with Y, J and H filters, for 121 s, 116 s, and 81 s, respectively (Carry, 2018). For a certain area of the sky, the aforementioned exposures are repeated four times, with small changes in telescope orientation in between them, so that the total observation time for the area is approximately one hour.

Although Euclid's main science goals are cosmological, it will also observe up to 150,000 solar system objects (Carry, 2018). Most of these are asteroids and they will appear in the Euclid images as streaks. The detection limit will be around 24.5 magnitudes for VIS, and 21 for analyzable spectral data. More detailed estimates of the observed objects are shown in Table 3.1. Finding and analyzing these asteroid streaks will make up the Euclid-SSO (Euclid Solar System Object) part of the mission. Euclid will mostly avoid the galactic and ecliptic planes, and focus on areas of sky that have galactic latitudes of more than 30°, and ecliptic latitudes of more than 15°, except for calibration fields, which will be closer to the ecliptic plane. For this reason, the asteroids detected by Euclid will largely be objects in high-inclination orbits. As Euclid will measure spectra of galaxies, in order to determine their redshifts, it will also collect spectral data of the asteroids, and radically increase the number of measured asteroid spectra in the infrared region. In addition to the spectral data, Euclid data can, in many cases, also be used to constrain several properties of the asteroids, such as the rotation period, spin orientation, and shape, as well as detect binary asteroids. Due to the relatively short observation time per asteroid, accurate orbit determination for the new previously unknown objects cannot be done on the

basis of Euclid data alone, but rough orbits and especially inclination distributions can be estimated. More accurate determination of orbits can be performed a posteriori. For example, the upcoming LSST will determine the orbits for a large number of asteroids, and this data can be used to retrace which objects were visible in the Euclid images during the observation time.

Since the data created by Euclid will consist of hundreds of thousands of images, taking several tens of petabytes of hard disk space, automated algorithms are necessary to analyze the data. Approximately 10 billion galaxies will appear in the data, of which upwards from 1 billion will be utilized for weak gravity lensing, and a close to 100 million galaxy redshifts will be measured [1].

3.2 Simulated Data

The simulated data is generated with Euclid Visible InStrument Python Package (VIS-PP), a Python program developed by Niemi (2015), with a special add-on to add the simulated trails of asteroids. The data mimics that of the Euclid VIS instrument. The NISP data is not simulated and analyzed in this work. The simulated data consists of images in FITS format with a width of 4096 and height of 4136 pixels, containing stars, galaxies, asteroid streaks, cosmic rays, noise and several different types of image artifacts. Corresponding ground truth data files are generated, which contain the coordinates of all stars, galaxies and asteroid streaks in the images. These single image files are a part of a larger 3x3 tile with a total of 9 images. Four of these 3x3 tiles are from the same region of the sky, with only minor dither movements between the exposures. In the real data, the dither movements are executed to avoid gaps in the data. There are gaps between the CCD sensors and the sensors might have dead pixels, so when the next exposure is taken in a slightly different orientation, the gaps and dead pixels do not always have the same

[1] Euclid Consortium, https://www.euclid-ec.org/

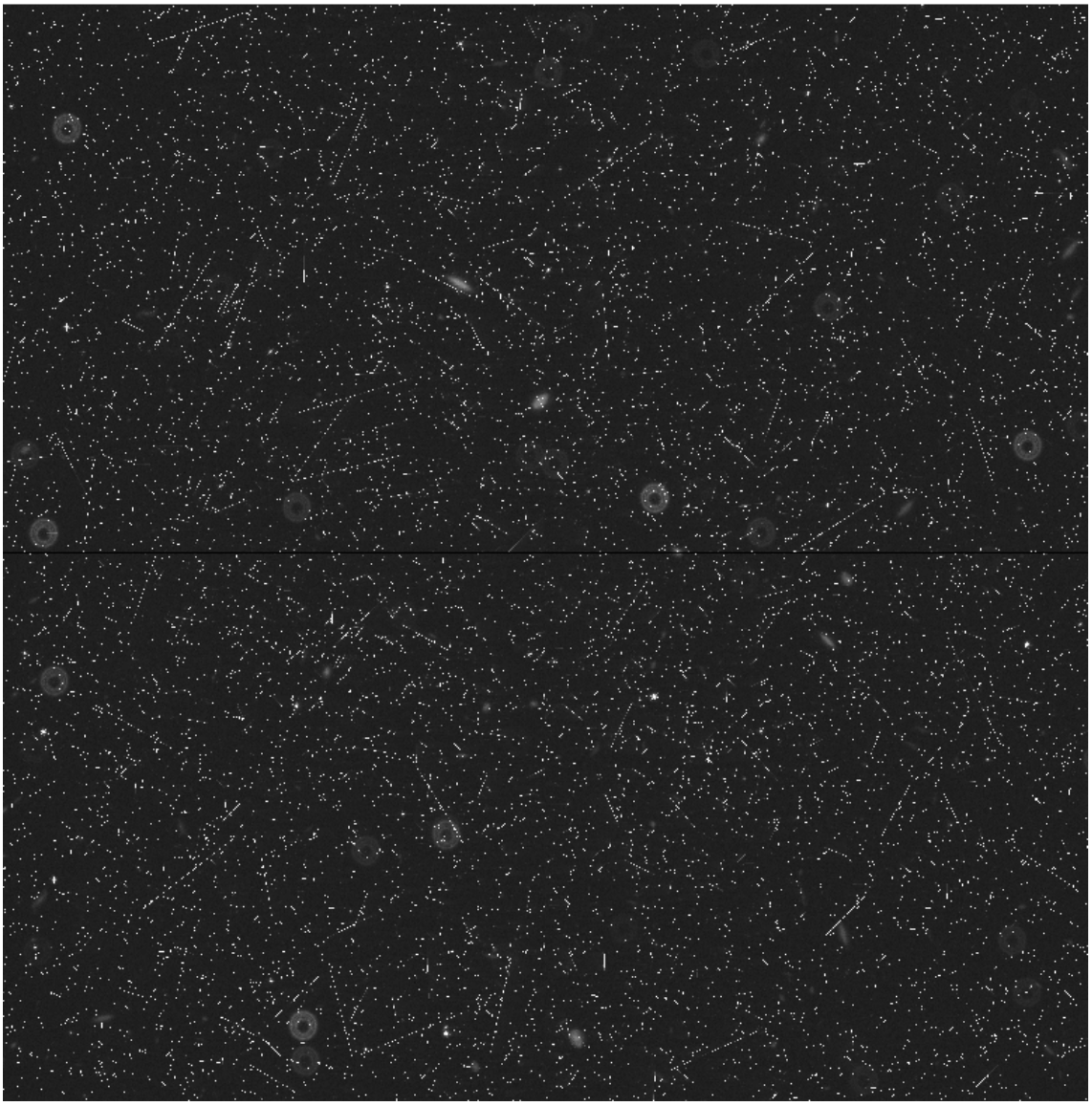

Figure 3.2: A FITS image generated with the VIS-PP program, visualized in logarithmic grayscale with SAOImage DS9 software. The size of the image is 4096×4136 pixels, corresponding to 409.6×413.6 arcseconds.

sky coordinates. These tiles can then be stacked to form a composite image with a total of 36 FITS files.

In the simulated data, the most distinct artifacts, i.e. image errors, are caused by cosmic rays. They are high energy radiation coming mostly from outside the Solar System, consisting mainly of high-energy protons and atomic nuclei. As these particles hit the CCD sensors, they cause the pixels to activate. In the images they

appear as bright streaks.

The data set used in this work consists of six 36-image stacks. The first stack contains asteroids with magnitudes between 20 and 21, the second stack with magnitudes from 21 to 22, and so on, until the final stack with magnitudes ranging from 25 to 26.

For StreakDet testing, the single FITS files are fed to the StreakDet pipeline, and then tiled, stacked and analyzed afterwards with a separate analysis program developed in Python. Examples of tiled and stacked images are shown in Section 4.2. For generating training data for machine learning, the FITS files are broken into smaller images, typically just a few pixels across, either clearly containing a part of an asteroid streak (positive training data), or containing no asteroids at all (negative training data).

Table 3.1: Estimates of Euclid survey parameters for different Solar System Object classes. NEA stands for near-Earth asteroid, MC for Mars-crossing, MB for main-belt, Trojan for Jovian Trojan, and KBO for Kuiper Belt Object. Euclid observations show the estimated total number of Solar System Objects in the data, while discoveries show the estimated number of previously unknown observed objects. The magnitude limits show the absolute magnitudes, for which the observation probability is 100%, 50% and 1%, respectively. "/h shows the sky motion of the objects, and pixels show the corresponding typical length of the streak in pixels. The data is from Carry (2018).

Population	Known population	Euclid discoveries	Euclid observations	Magnitude limits	"/h	Pixels
NEA	16062	$1.4^{+1.0}_{-0.5}\times10^4$	$1.5^{+1.0}_{-0.6}\times10^4$	22.75 23.75 26.50	$43.3^{+36.5}_{-19.9}$	67.9
MC	15488	$1.0^{+1.7}_{-0.8}\times10^4$	$1.2^{+1.7}_{-0.8}\times10^4$	21.00 21.25 22.75	$41.3^{+22.6}_{-14.9}$	64.8
MB	674981	$8.2^{+2.5}_{-2.2}\times10^4$	$9.7^{+2.5}_{-2.2}\times10^4$	19.50 20.00 21.25	$32.5^{+7.9}_{-5.5}$	51.0
Trojan	6762	$7.1^{+9.3}_{-4.9}\times10^3$	$7.5^{+9.5}_{-5.0}\times10^3$	17.00 17.25 18.25	$13.3^{+1.4}_{-1.1}$	20.9
Centaur	470	$2.2^{+2.1}_{-1.4}\times10^3$	$2.2^{+2.1}_{-1.4}\times10^3$	14.75 15.50 18.25	$4.0^{+2.9}_{-1.5}$	6.2
KBO	2331	$5.3^{+1.6}_{-1.3}\times10^3$	$5.5^{+1.6}_{-1.3}\times10^3$	8.25 8.75 10.00	$0.6^{+0.3}_{-0.1}$	1.0
Comet	1301	$21.5^{+4.2}_{-3.6}$	$38.2^{+4.9}_{-4.3}$	18.25 19.00 22.00	$4.4^{+6.2}_{-1.8}$	6.9

4. StreakDet

StreakDet (streak detection and astrometric reduction), created by Virtanen et al. (2016) is an ESA-funded software developed to detect and analyze object trails from optical observation data. StreakDet has been developed mainly to detect space debris either from Earth-based or space-based platforms, and it can detect long, faint and also curved streaks. Its focus is in being able to detect streaks from single images, in contrast to finding consecutive streaks from stacked images, a task for which it is not well optimized.

The initial tests run by the StreakDet developers gave detection sensitivities of about 90% for bright streaks (signal-to-noise ratios of >1), and about 50% for dimmer streaks (SNR = 0.5).

4.1 Program Pipeline of StreakDet

The StreakDet pipeline consists of three main phases: segmentation, classification and lastly astrometric and photometric reduction. The following descriptions of the algorithms are summarized from Virtanen et al. (2016).

The segmentation step converts the analyzed image into a black and white (BW) image with a color depth of 1 bit. In other words, after segmentation, every pixel has either a value of 0 or 1. The idea is to make the following steps computationally less demanding. The BW image is created with the aid of two gray-scale mean-filters. The first filter uses a smaller area, for example 3x3 pixels, for which

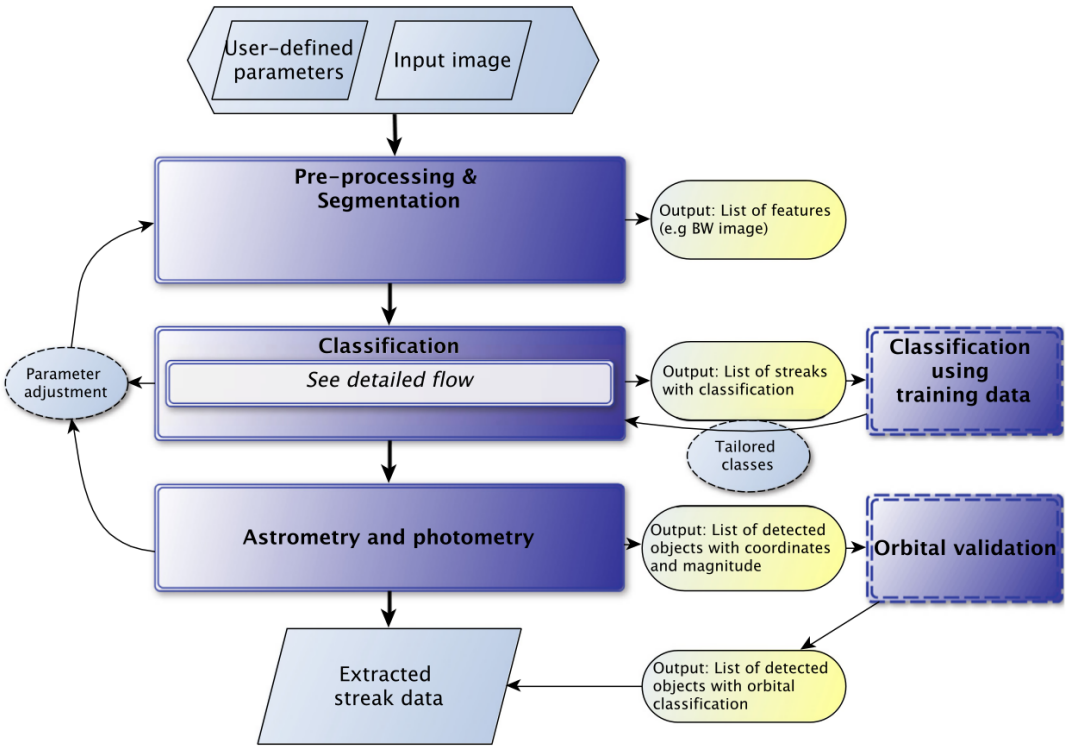

Figure 4.1: A visualization of the StreakDet program pipeline. From Virtanen et al. (2016).

it calculates a mean pixel value. The second filter uses a larger area, e.g., 21x21 pixels, and calculates the mean pixel value for that area. The BW image is created by subtracting the differences of the means calculated by the grayscale mean-filters. The idea behind the mean-differences is to detect groups of pixels whose value differs from the background. Because the mean-difference calculation is locally executed, it is typically not biased by global background gradients, which reduces the need for preprocessing the image before feeding it to StreakDet. Before segmentation, the star density of the image is calculated or manually set, in order to determine the proportion of white pixels to black pixels in the segmented image.

After the segmentation has turned the image to black and white, filtering processes are applied, which aim to remove non-streak-like features from the pipeline. An adapted version of binary erosion is used, which gets rid of isolated active pixels and keeps pixels which are part of a larger structure, such as a streak. Then a

reconstruction filter is used to strengthen the remaining features. After the previous steps have removed most of the noise and small stars, larger stars are removed by multiple-window pixel-density evaluation, which removes pixel groups whose number of active pixels does not grow linearly when the window size is increased. Finally, the remaining features and their properties are indexed into a list with a Connected Component Labeling (CCL) algorithm.

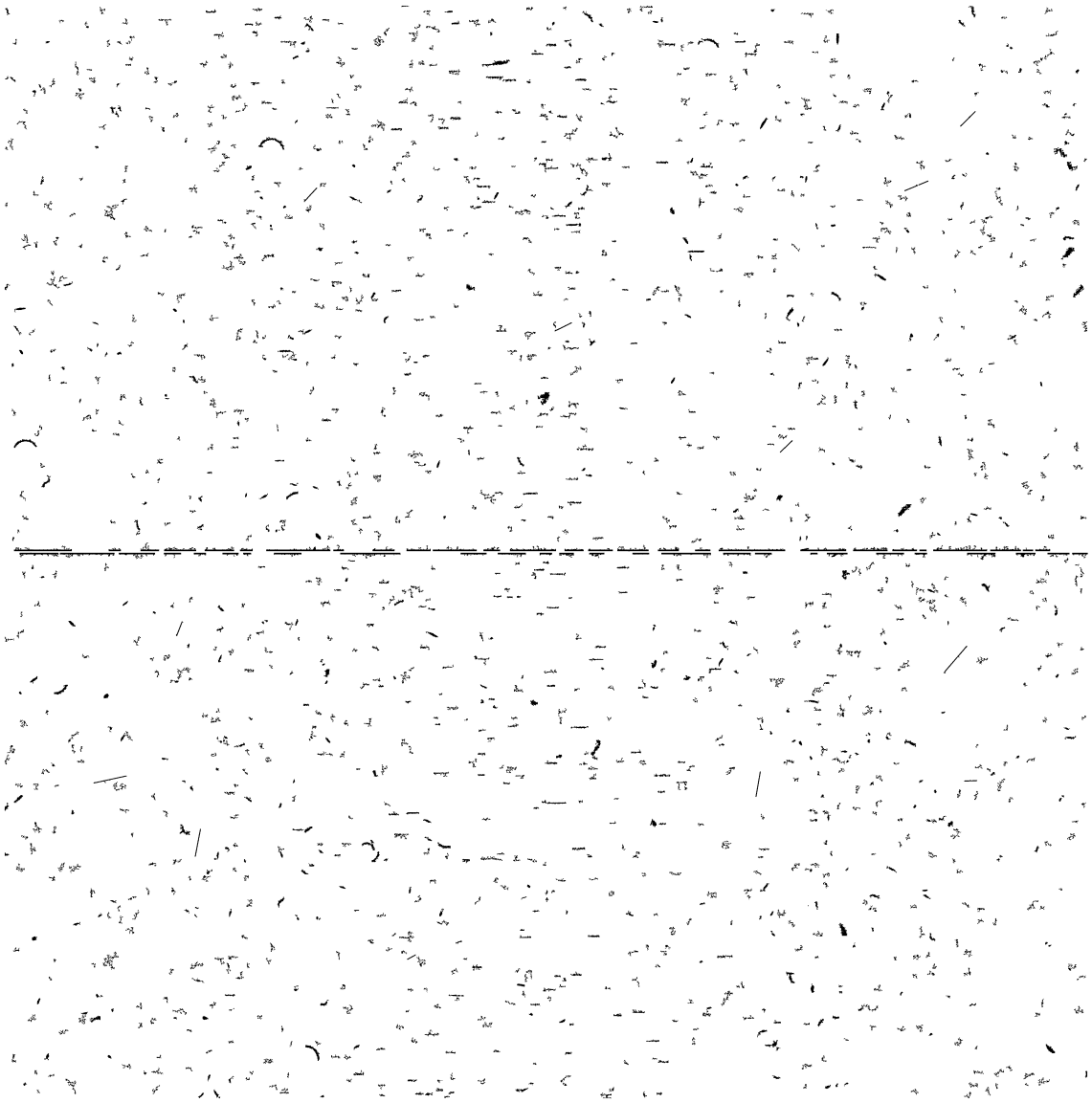

Figure 4.2: An example of an image after the segmentation phase of the StreakDet pipeline, when applied to one of the FITS files of simulated Euclid data used in this work.

After segmentation, the CCL features are inputted into the classification phase, which consists of the following three steps:

1. Characterization of BW CCL features.

2. Characterization of BW CCL features that correspond to streaks.

3. Characterization of original grayscale features that correspond to streaks.

Each of the steps uses classification processes to find the streaks, and filtering processes to get rid of the non-streaks. During each step, eigenvalue analysis is used to compute feature parameters such as width, orientation angle, aspect ratio, curvature, and porosity (referring to the compactness of the feature). Both linking and unlinking are done during the BW steps. Linking implies connecting found features that are likely to be parts of the same longer streak, whereas unlinking implies dividing a large found feature into smaller ones, if it seems likely that the sub-features are actually separate streaks. During the grayscale parametrization, point-spread-function (PSF) fitting with a moving 2D Gaussian approach, utilizing the non-linear Levenberg-Marquardt least-squares method, is used to refine the streak parameters. The grayscale parametrization is done by starting with the BW features and then finding the parameters of the corresponding grayscale streak with the aforementioned algorithms. The grayscale fitting extends also outside the BW bounding box, in case only a part of the real streak was found during the BW process. Finally, the grayscale features are classified and filtered according to their PSF width and curvature. During the classification phase, an optional step is to use the k-nearest-neighbors algorithm with principal-component analysis (PCA) to help classify the identified features into streaks and non-streaks.

The final, optional, phase of the StreakDet pipeline is astrometric and photometric reduction, during which the streak parameters are converted into final output parameters, such as sky coordinates and magnitude. The sky coordinates

of the streaks are found by linear mapping and polynomial fit with a number of the field stars. The magnitudes of the streaks are calculated with the aid of USNO CCD Astrograph Catalogue 4 (UCAC4), comparing the magnitudes of the UCAC4 stars to the pixel values of the stars in the analyzed image, and then calculating the corresponding magnitudes for the streak objects.

StreakDet is programmed in C++, and it uses a few external libraries, such as OpenCV, CFITSIO, LMFit, libssrckdtree, Boost, JsonBox and the UCAC4 star catalogue. The PSF fitting stage can be run in parallel with several CPU cores, but otherwise StreakDet is not yet parallelized, so it uses one CPU core at a time. StreakDet is run from the command line, although it has also a prototype version of a web-based user interface. The images fed to StreakDet have to be provided in the FITS format, and contain several parameters in the header part of the file, such as sky coordinates of the image, scaling of the image, observation date, and exposure time. StreakDet has options for normal and full output modes. For normal mode, it outputs only the final streak parameters and the astrometric results (if astrometry is set on in the settings). In full mode, the program outputs results also from the intermediary stages of the pipeline, namely the CCL, pre-PSF, post-PSF and final results. The output parameters are saved into CSV files, and the software also provides visual images of the detected streaks.

4.2 Analysis Software

In order to compare the results given by StreakDet to the ground truth, we developed a test and analysis software in Python, consisting of approximately 1500 lines of code. StreakDet outputs simple CSV files containing the coordinates, lengths and angles of the streaks found from several points of the StreakDet pipeline. These need to be compared to the true properties of the streaks, known from the ground truth files generated simultaneously with the simulated Euclid data. The analysis

program can compare the ground-truth streaks to the streaks found by StreakDet, and compute statistics of the hits and misses. It can also plot the FITS images together with markings of the locations of the true streaks and the streaks found by StreakDet.

StreakDet was tested by running StreakDet on single FITS files, and only afterwards combining and stacking all the data by the separate Python analysis program. The reason for this is that StreakDet itself did not appear to work well with stacked images. Also, StreakDet did not scale linearly with increasing file size. Running StreakDet on one FITS file of approximately 16 Mpix (4k×4k) took around 1-2 minutes, depending on settings. Running StreakDet on a tiled 3x3 image of around 170 Mpix (13k×13k) took around 20-60 minutes, which suggests a scaling of $O(n \log n)$ or worse. For these reasons, the 4k×4k images are run one by one with StreakDet, and then the images, StreakDet results, and ground-truth data are tiled and stacked afterwards with the analysis program. Also, because StreakDet uses only 1 CPU most of the time, N StreakDet processes can be run in parallel with a computer with N CPU cores, which makes it possible to run StreakDet on 36 FITS images in only approximately $\max(1, 36/N)$ times the time it takes to analyze one FITS image.

The analysis program can analyze the true positives (hits), false negatives (misses), and false positives for any single FITS image that has gone through StreakDet. It can also tile the FITS images, StreakDet results and ground truth for a full tile of 3x3 FITS images, and then analyze the hits, misses, and false positives for all those at once. Furthermore, it can stack the images, results, and ground truth for all 4 dithers from a total of 36 separate images (4 dithers x 3x3-sized tiles). The program can also plot the number of finds for streaks of different lengths, and the number of finds for different magnitudes. In addition, the number of false positives can be plotted as a histogram as a function of streak length.

To get rid of most of the false positives, we developed the so-called multistreak approach and implemented it in the Python analysis program. As the data has images from four dithers, the asteroids appear as multiple line segments along the same line in the stacked image (see Figure 4.3).

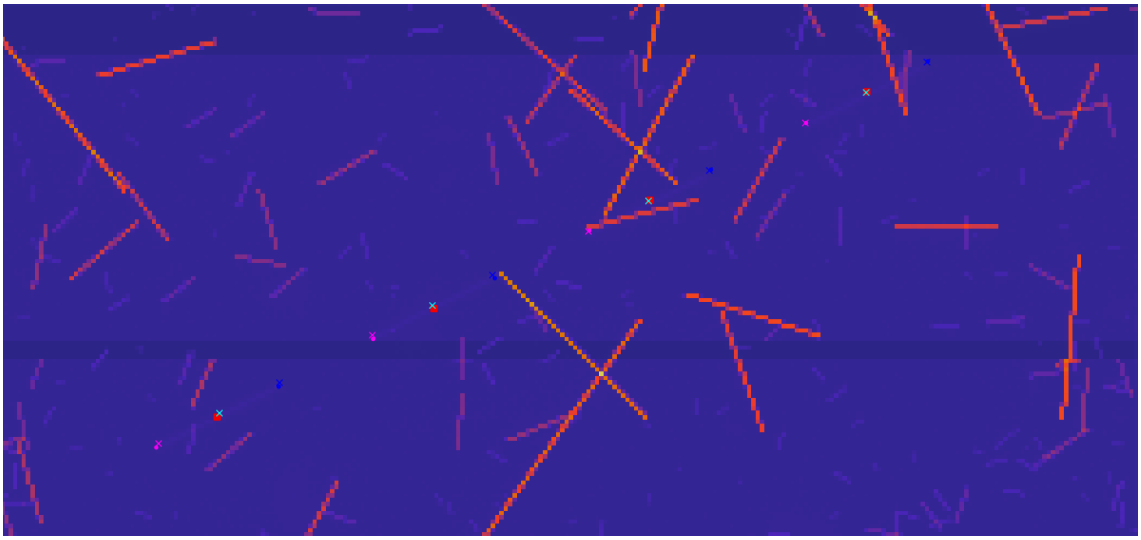

Figure 4.3: An example of a multistreak in a stacked image, consisting of four separate 30-pixel-long streaks (sky motion of 20 "/h) from different dithers, created by the same asteroid.. The squares mark the ground truth streaks, and the crosses mark the finds by StreakDet. In this case, StreakDet managed to find all four single streaks of the multistreak. The surrounding brighter streaks are cosmic rays. The width of the image is approximately 300 pixels (30 arcseconds) and the height is roughly 140 pixels (14 arcseconds).

The multistreak pipeline goes as follows:

1. Run StreakDet separately on all 36 FITS images of all four dithers.
2. Tile and stack the FITS images, StreakDet results and ground truth data with Python analysis program.
3. Search for ground truth multistreaks that have 2-4 streaks along the same line (each streak in different dither).
4. Search for StreakDet multistreaks that have 2-4 streaks along the same line (each streak in different dither).

5. Analyze which StreakDet multistreaks match to ground truth multistreaks and which do not. To be classified as a match, at least two single streaks in a StreakDet multistreak have to match to those of the ground truth multistreak.

All the aforementioned analysis, including single-streak and multistreak analysis, can be done for data from four different points of the StreakDet pipeline: after segmentation, before PSF fitting, after PSF fitting, and for final results.

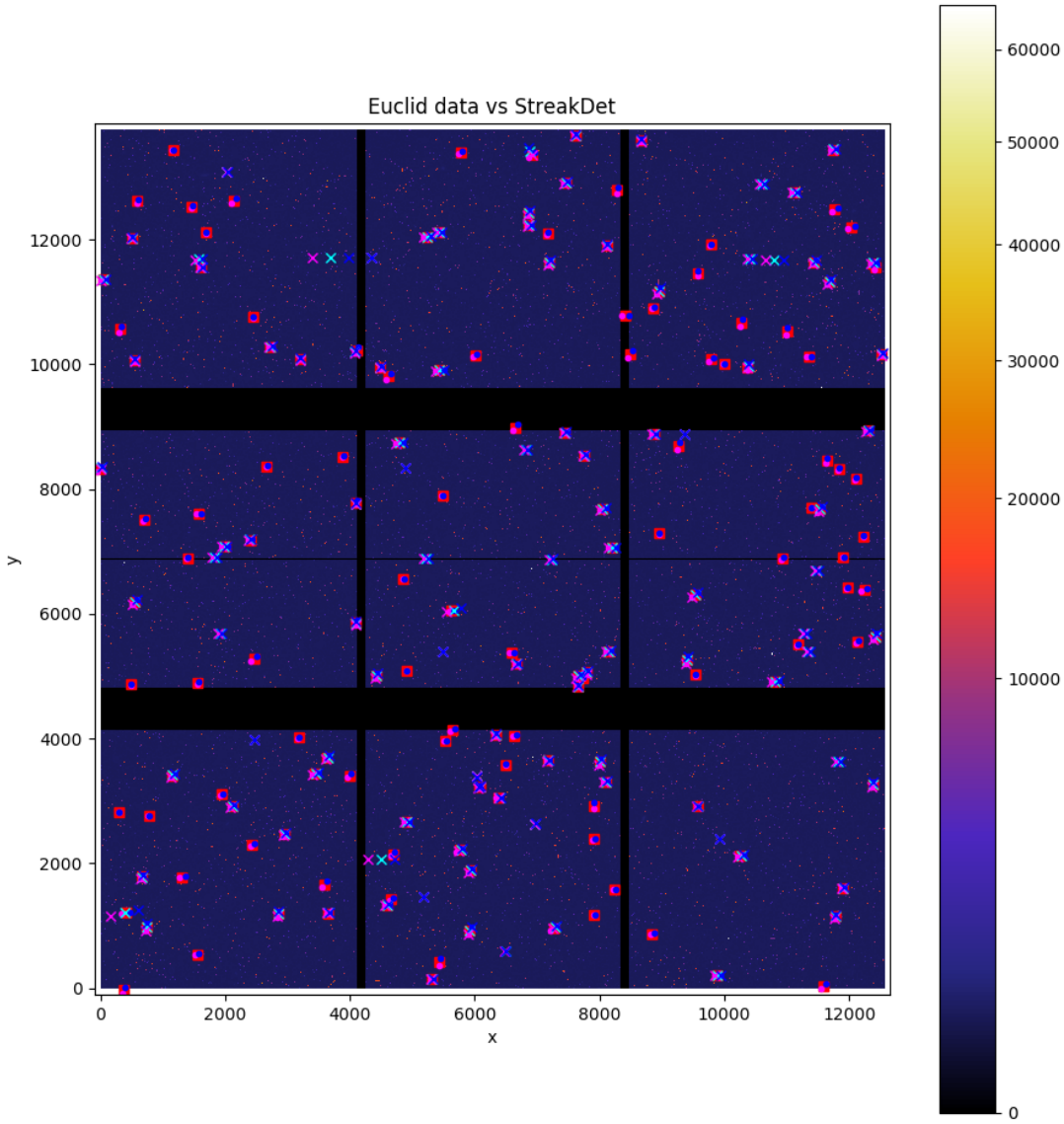

Figure 4.4: An example of a tiled image, which contains all 9 FITS images of a single dither, and the corresponding ground truth and StreakDet data. The squares are ground truth streaks, and the crosses are StreakDet finds. In the simulated data there are ground truth streaks also outside the visible areas of the FITS images (i.e., outside the tiled image, or on the black areas between the separate images), but they are ignored in the analysis, because it is impossible for StreakDet to find them since they are not visible in the images. The color scale corresponds directly to pixel values in the FITS images.

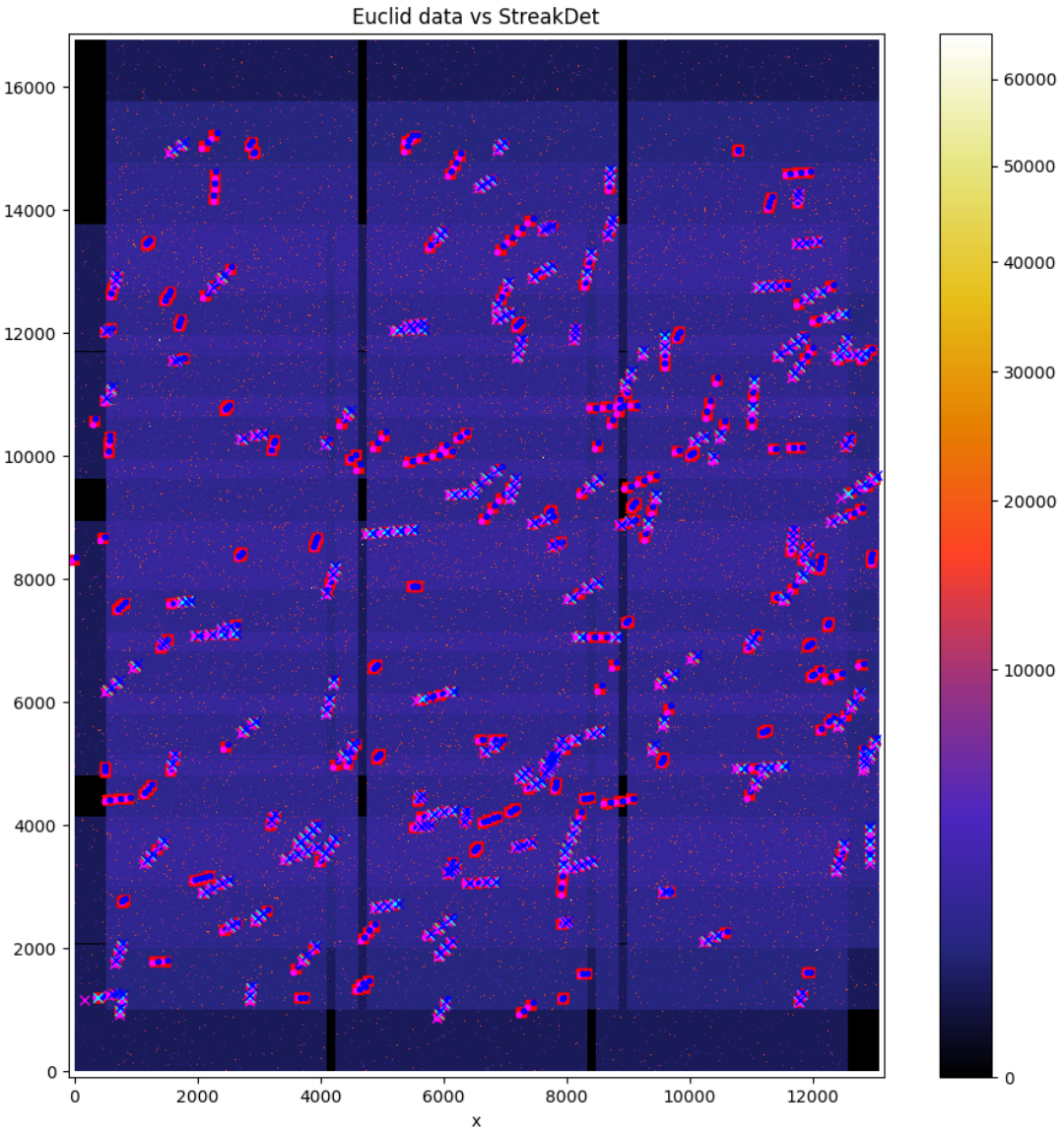

Figure 4.5: An example of a stacked image, containing all 36 FITS images of all four dithers, plus all the corresponding ground truth and StreakDet data. This image is after the multistreak analysis, which has removed all "lonely" ground-truth and StreakDet streaks, leaving only the multistreaks. The color scale corresponds directly to pixel values in the FITS images.

5. Machine Learning

Machine learning is a form of artificial intelligence, where the program learns from data, instead of having explicitly programmed rules or algorithms. In recent years, both because of more powerful processors and breakthroughs in algorithms, machine learning, especially so-called deep learning, has been used to reach remarkable results in many areas that have previously been very difficult for computers to handle.

5.1 Basics of Machine Learning

Artificial intelligence (AI) is an old field of research, generally recognized as having started with an article by McCulloch and Pitts (1943), who first suggested the possibility of creating artificial neural networks. Other notable early work, leaning more towards the philosophy of artificial intelligence, was the concept to be known as the Turing test, formulated by Turing (1950). Artificial intelligence as a term was coined for the Dartmouth Summer Research Project on Artificial Intelligence, held in 1956. The proposal for the summer project (McCarthy et al., 1955) stated:

> We propose that a 2 month, 10 man study of artificial intelligence be carried out during the summer of 1956 at Dartmouth College in Hanover, New Hampshire. The study is to proceed on the basis of the conjecture that every aspect of learning or any other feature of intelligence can in principle be so precisely described that a machine can be made to simulate it. An attempt will be made to find how to make machines

> use language, form abstractions and concepts, solve kinds of problems now reserved for humans, and improve themselves. We think that a significant advance can be made in one or more of these problems if a carefully selected group of scientists work on it together for a summer.

Unfortunately, despite the early optimism, solving artificial intelligence has turned out to take slightly more time than a single summer, but at least the foundations of the research field were laid in 1956.

In the old days, AI was approached mostly with logic-based programming, known also as GOFAI (Good Old-Fashioned Artificial Intelligence), slowly evolving to probability-based algorithms, and to the currently most-used method, machine learning (Russell and Norvig, 2009). As a term, artificial intelligence is not unequivocally defined, but generally it is taken to mean intelligent and rational behavior exercised by a computer program or other non-natural agent, in order for it to maximize the chances of achieving its goals.

As a sub-field of AI, machine learning means algorithms that can learn from data, instead of being explicitly programmed. Roughly speaking, machine learning tasks can be divided into two categories, supervised and unsupervised learning. Supervised learning means that the algorithm is trained with labeled data. As an example, in image recognition, supervised learning means showing the algorithm images, and also telling it what each image represents. Supervised machine learning applications are, for example, classification, regression, and learning to perform a task via reinforcement learning. An example of classification is learning to recognize handwritten letters in order to convert handwritten text into typed text. An example of regression is learning to predict the price of a house, when given input features such as the size, age and location of the house. An example of reinforcement learning is when an AI algorithm learns to play a computer game by trial and error.

Unsupervised learning refers to methods where the training data is unlabeled.

The goal of unsupervised learning is typically to find patterns and structures in the input data. Applications for unsupervised machine learning methods are clustering, dimension reduction and anomaly detection, for example. An example of clustering is grouping people into smaller communities by the properties of their data, when analyzing large social networks, such as Facebook. An example of dimension reduction is simplifying some high-dimensional data, such as DNA data, for easier analysis and comparison. Anomaly detection can be used, for example, to search for atypical transactions in monetary systems, indicative of fraud.

There is a huge number of different machine learning algorithms, and many of them can be used for several, if not all of the aforementioned tasks and applications. One of the most versatile and powerful algorithms has turned out to be deep neural networks.

5.2 Deep Learning

Although it were neural networks that in some sense started the whole field of AI, their real breakthrough has happened only in the last few years, as it has become possible to train very deep and large neural networks. Utilizing these deep networks has become known as deep learning. During recent years, deep learning methods have reached important milestones in many areas, such as image recognition, speech recognition, natural language processing, and game-playing (LeCun et al., 2015). The breakthroughs have been powered both by enhancements in algorithms, and by increases in computing power, with the exploitation of GPUs (Graphics Processing Unit) and even ASICs (Application-Specific Integrated Circuit) such as Google's TPUs (Tensor Processing Unit).

The following sections explaining the principles and mathematics of logistic regression and neural networks are adapted from the online courses by Andrew Ng[1].

[1] https://www.deeplearning.ai/

5.2.1 Training, cross-validation and test sets

In order for a machine learning algorithm to work, some sort of training data is needed. Moreover, typically the training data has to be in a certain, consistent format. For example, for an image-recognition project, the data sets can be constructed the following way.

A digital image can be thought of as a matrix, whose every element corresponds to a pixel and has a numerical value. In color images, there are typically three separate matrices, containing the pixel values for red, green, and blue (RGB). In grayscale images, such as the FITS images used in this work, there is only one matrix. For machine learning purposes, the matrix is usually flattened into a feature vector $\mathbf{x}$, which contains all the pixel values of an image in a single column. For example, for a grayscale image with the size of 32×32 pixels, the feature vector $\mathbf{x}$ is a column vector with dimensions of 1024×1. For an RGB image of the same size, the feature vector has a size of 3072×1 (32×32×3). The range of values in the feature vector, in this case the values of pixels, is usually normalized to range from 0 to 1.

In machine learning projects, there are usually training, cross-validation and test sets. All the sets consists of a number of $(\mathbf{x}, y)$ pairs, where $\mathbf{x}$ is a feature vector with the dimension n_x, and y is its label. In binary classification, the label y is either 0 or 1. For example, if the data consists of asteroid and non-asteroid images, an image with an asteroid in it has a label of 1, whereas an image without an asteroid has a label of 0. In other words, a training example for binary classification is a $(\mathbf{x}, y)$ pair, where $\mathbf{x} \in \mathbb{R}^{n_x}$ and $y \in \{0, 1\}$. A training set consists of m training examples, i.e., $\{(\mathbf{x}^{(1)}, y^{(1)}), (\mathbf{x}^{(2)}, y^{(2)}), \ldots , (\mathbf{x}^{(m)}, y^{(m)})\}$. In non-binary classification the labels correspond to defined, numbered classes. For example, when teaching an algorithm to recognize handwritten numbers, the label for each image is a number from 0 to 9.

The training set is used to train the machine learning algorithm, and cross-

validation set is used to see how well a trained model can classify images it has not directly learned from. Typically the algorithm's hyperparameters, such as learning rate or neural network structure, are optimized in several iterations of training the model and testing it with the cross-validation set. Finally, when the model outputs results with desired accuracy with the cross-validation data, it is tested on a separate test set, consisting of images that it has not seen before. This three-step procedure is due to the fact that often the machine learning model is indirectly overfitted to the cross-validation set through the many optimization iterations, even though the algorithm directly learns only from the training set.

5.2.2 Logistic regression

Logistic regression is not deep learning per se, but it can be a helpful stepping stone before implementing actual deep learning algorithms. Logistic regression is an old statistical regression model, developed by Cox (1958). It can be adapted into a simple machine learning model, and with the following implementation, it can be thought of as a neural network with only one neuron. Logistic regression can be a helpful model in the beginning of a machine learning project, because it is fairly simple and easy to debug, and thus it can be used as a sanity check before moving to more advanced algorithms.

The idea behind logistic regression is that when the algorithm is given $\mathbf{x}$, it returns $\hat{y} = P(y = 1 \mid \mathbf{x})$. For example, given an image, it returns the probability that the image contains an asteroid. The label estimate $\hat{y}$ is sometimes marked also as a, as for activation. The parameters for logistic regression algorithm are the weights $\mathbf{w}$ and bias b. The weights $\mathbf{w}$ are placed in a column vector with the same dimension as $\mathbf{x}$, so that $\mathbf{w} \in \mathbb{R}^{n_x}$, and the bias b is a scalar, so $b \in \mathbb{R}$.

The basic activation function for logistic regression, for a single training ex-

ample, can be written as

$$\hat{y} = a = \sigma(z)\,, \tag{5.1}$$

where σ is the sigmoid function, defined as

$$\sigma(z) = \frac{1}{1+e^{-z}}\,, \tag{5.2}$$

and z is defined as

$$z = \mathbf{w}^T\mathbf{x} + b\,. \tag{5.3}$$

The sigmoid function is used, because it always returns a number between 0 and 1, and can be directly used to output a probability. Thus, the machine learning process of the logistic regression is to find such values for $\mathbf{w}$ and b that a (or $\hat{y}$) returns a good estimate for the label of a given image $\mathbf{x}$. For logistic regression, the initial values of $\mathbf{w}$ and b are typically set to zero.

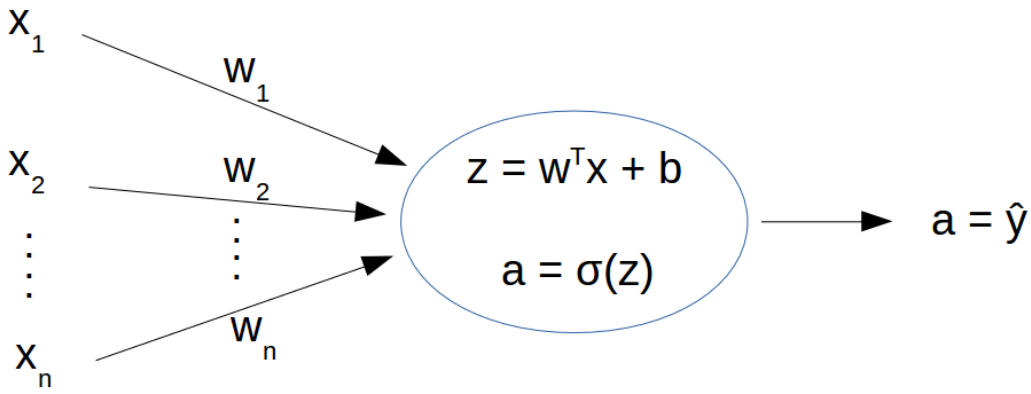

Figure 5.1: A visualization of the logistic regression as a neural network with a single neuron. The x_i mark the elements of the feature vector $\mathbf{x}$. In the case of image recognition, they correspond to the pixels of the image. The w_i are the weights given to the values of the pixels. The neuron then activates according to the input $\mathbf{x}$, weights $\mathbf{w}$ and bias b, returning a.

In the learning stage of the machine learning process, some kind of a loss function is needed, in order to optimize $\mathbf{w}$ and b and to cause the estimates a to

approach the true labels y. A reasonable loss function (also known as error function) for the purposes of logistic regression can be defined as

$$L(a, y) = -[y \log a + (1 - y) \log(1 - a)] . \tag{5.4}$$

The reasoning behind this loss function is that if $y = 1$, the loss L approaches 0 as a approaches 1, and in turn, if $y = 0$, the loss L approaches 0 as a approaches 0. In addition, it is an easily optimizable convex function. The loss function computes the error between the true and estimated labels of a single training example, so the error of the whole training set can be calculated by taking the mean loss over all the training examples with the cost function

$$J(\mathbf{w}, b) = \frac{1}{m} \sum_{i=1}^{m} L(a^{(i)}, y^{(i)}) . \tag{5.5}$$

The previous steps make up the forward propagation part of the algorithm, i.e., starting from the training examples and propagating through the steps to calculate the estimated labels and the cost.

The next phase is the backward propagation. The cost function J can be minimized by using stochastic gradient descent, which means calculating the derivatives of $\mathbf{w}$ and b with respect to J, taking a small step "downhill" towards the minimum of the cost function, and iterating until the minimum, or a value close to it, is reached. So,

$$\mathbf{w} = \mathbf{w}_{previous} - \alpha \frac{\partial J(\mathbf{w}, b)}{\partial \mathbf{w}} , \qquad b = b_{previous} - \alpha \frac{\partial J(\mathbf{w}, b)}{\partial b} , \tag{5.6}$$

where α is the learning rate, the size of the downhill step.

The derivatives of the cost function J relative to $\mathbf{w}$ and b can be calculated step by step with the help of the chain rule, moving back the aforementioned stages:

$$\frac{\partial L(a, y)}{\partial a} = -\frac{y}{a} + \frac{1 - y}{1 - a} \tag{5.7}$$

$$\frac{\partial L(a, y)}{\partial z} = \frac{\partial L(a, y)}{\partial a} \frac{\partial a}{\partial z} = (-\frac{y}{a} + \frac{1 - y}{1 - a}) a(1 - a) = a - y \tag{5.8}$$

Next, it can be seen from Equation (5.3) that in the case of a single training example, $\frac{\partial J(\mathbf{w},b)}{\partial b} = \frac{\partial J(\mathbf{w},b)}{\partial z}$. Similarly, for a weight w_j (an element in vector $\mathbf{w}$) corresponding to an input feature x_j (an element in feature vector $\mathbf{x}$), the derivative is $\frac{\partial J(\mathbf{w},b)}{\partial w_j} = x_j \frac{\partial J(\mathbf{w},b)}{\partial z}$. Thus, for all the weights $\mathbf{w}$, the derivative is $\frac{\partial J(\mathbf{w},b)}{\partial \mathbf{w}} = \mathbf{x} \frac{\partial J(\mathbf{w},b)}{\partial z}$. The corresponding derivatives for the whole training set are calculated as the means of these single-training-example derivatives:

$$\frac{\partial J(\mathbf{w},b)}{\partial b} = \frac{1}{m}\sum_{i=1}^{m} \frac{\partial L(a,y)}{\partial z}^{(i)} = \frac{1}{m}\sum_{i=1}^{m}(a^{(i)} - y^{(i)}) \tag{5.9}$$

$$\frac{\partial J(\mathbf{w},b)}{\partial \mathbf{w}} = \frac{1}{m}\sum_{i=1}^{m} \mathbf{x}^{(i)} \frac{\partial J(\mathbf{w},b)}{\partial z}^{(i)} \tag{5.10}$$

The calculated derivatives can then be inserted into Equations (5.6), which concludes one cycle of forward and backward propagation. This cycle is repeated until the cost function is optimized.

To avoid going through the training set one image or other example at a time with a for-loop, the algorithm can be vectorized by placing the training examples into matrices. This makes it possible to teach the algorithm with numerous training examples or even with the whole training set simultaneously. This can decrease the computing time of the algorithm by several orders of magnitude. The $(\mathbf{x}, y)$ training example pairs can be split into matrices X (with the size of $n_x \times m$, in other words, $X \in \mathbb{R}^{n_x \times m}$) and Y (with the size of $1 \times m$, in other words, $Y \in \mathbb{R}^{1 \times m}$):

$$X = \begin{bmatrix} | & | & & | \\ \mathbf{x}^{(1)} & \mathbf{x}^{(2)} & \cdots & \mathbf{x}^{(m)} \\ | & | & & | \end{bmatrix}, \qquad Y = \begin{bmatrix} y^{(1)} & y^{(2)} & \cdots & y^{(m)} \end{bmatrix}.$$

Using the vectorized approach, the activation function for logistic regression with m training examples becomes as follows:

$$\hat{Y} = A = \sigma(Z), \tag{5.11}$$

where $A = \begin{bmatrix} a^{(1)} & a^{(2)} & \dots & a^{(m)} \end{bmatrix}$ and $Z = \begin{bmatrix} z^{(1)} & z^{(2)} & \dots & z^{(m)} \end{bmatrix}$. The sigmoid function σ is vectorized as well, i.e., applied element-wise, so that when vector Z is fed into it, it returns the vector A. The formula to calculate Z becomes

$$Z = \mathbf{w}^T X + \mathbf{b}\,. \tag{5.12}$$

where $\mathbf{w}$ is the same column vector as in the non-vectorized case, X is the matrix shown above and $\mathbf{b}$ is a row vector with m elements, and every element is the same scalar b.

The loss function can be vectorized, whereas the cost function was already defined for the whole training set, and therefore stays the same. The derivatives can be vectorized the following way:

$$\frac{\partial L(A,Y)}{\partial Z} = A - Y \tag{5.13}$$

and thus $\frac{\partial L(A,Y)}{\partial Z} = \begin{bmatrix} \frac{\partial L(A,Y)}{\partial z}^{(1)} & \frac{\partial L(A,Y)}{\partial z}^{(2)} & \dots & \frac{\partial L(A,Y)}{\partial z}^{(m)} \end{bmatrix}$. This can be used to calculate the derivatives of cost function L with respect to bias b and weights $\mathbf{w}$, and we end up with vectorized versions of Equations 5.9 and 5.10:

$$\frac{\partial J(\mathbf{w},b)}{\partial b} = \frac{1}{m}\sum_{i=1}^{m} \frac{\partial L(A,Y)}{\partial z}^{(i)} \tag{5.14}$$

$$\frac{\partial J(\mathbf{w},b)}{\partial \mathbf{w}} = \frac{1}{m} X \frac{\partial L(A,Y)}{\partial Z}^{T} \tag{5.15}$$

After the the logistic regression algorithm has been trained, the weights $\mathbf{w}$ and bias b can be used to predict labels for the test set or other data using the Equations (5.1), (5.2), and (5.3).

5.2.3 Deep neural networks

Artificial neural networks are inspired by natural neural networks, also known as brains. Brains with multiple neurons with connections between them work better

than brains with a single neuron, and this remark holds true for artificial neural networks as well. The previously outlined implementation of logistic regression can be thought of as a neural network consisting of one neuron. An actual artificial neural network can then be built by combining several of these neurons into a network. A neural network typically has the neurons organized into layers. The beginning of the network, where the input data $\mathbf{x}$ is fed into the system, is called the input layer. In logistic regression, directly after the input layer comes the output layer. In neural networks, between the input and output layers there are so-called hidden layers. The input layer is typically not considered an actual layer, because the first calculations and activations are only performed in the next layer.

The depth of a neural network can be examined with the credit assignment path (CAP), which is defined as the number of hidden layers plus one (the output layer), or in other words, the number of all layers excluding the input layer. A neural network with a CAP of 4 is then said to be a 4-layer neural network. There is no exact definition that differentiates shallow neural networks from deep ones, but typically it is considered that deep neural networks have a CAP of larger than two, i.e. they have more than one hidden layers. A network with only one hidden layer is then considered a shallow neural network.

The reason to use deep neural networks instead of shallow ones, without going into the exact mathematics, is that a relatively small multilayer network can learn complex functions that would require a much larger number of neurons in a network with only one hidden layer. In other words, theoretically a shallow neural network with a CAP of 2 could be trained to learn almost any function, given a large enough number of neurons in the hidden layer, but in practice a multilayer network can learn the same functions with a lot less neurons and computing time. At least for certain functions, the number of neurons needed to learn a function with n input features in the feature vector $\mathbf{x}$ scales as $O(2^n)$ for a shallow neural network, whereas

for a deep neural network the number of neurons needed scales only as $O(\log n)$.

In a deep learning project, the optimizable parameters are the weights W and bias b. Additionally, there are so-called hyperparameters, such as the learning rate α, the number of gradient descent iterations, the choice of the activation functions for the neurons, the number of layers in the neural network, and the number of neurons in each layer. Each of these hyperparameters affects how well the algorithm learns, and optimizing them is more of an art than a science.

For logistic regression, the values of $\mathbf{w}$ and b could be set to zero before starting the learning process. Although setting b to zeros works for neural networks as well, it turns out that it does not work for W. The reason is that if all the neurons in a layer have identical initial values for the weights $\mathbf{w}$, the gradient descent updates them always similarly to each other, and their activations will stay identical to each other. It can be shown that a layer with identical weights for every neuron corresponds to a layer with only one neuron, and thus cannot learn complex functions. For this reason, the initial values for W have to be set randomly, with different values for each neuron.

For notation, L is used for marking the number of layers, i.e., the CAP. The number of neurons, or units, in layer l is marked by $n^{[l]}$, so if the first hidden layer consists of 10 neurons, $n^{[1]} = 10$. Similarly, the activations a and their subfunctions z and g are marked by layers, so that $a^{[l]} = g^{[l]}(z^{[l]})$, and $z^{[l]}$ is calculated by using the weights $\mathbf{w}^{[l]}$ and bias $b^{[l]}$ of that layer, similarly as in Equation (5.3). The g is used as a symbol for the activation function. In the case of logistic regression, the activation function was the sigmoid function σ. It turns out that the sigmoid function works well for the output layer in binary classification projects, but other functions give better results for the neurons in the hidden units. Nowadays, one of the most typical activation function for the hidden layers is a rectified linear unit

(ReLU) (Glorot et al., 2011), which is defined as

$$a = \max(0, z)\,. \tag{5.16}$$

The ReLU function simply returns the value of z, when z is positive, or 0, when z is negative.

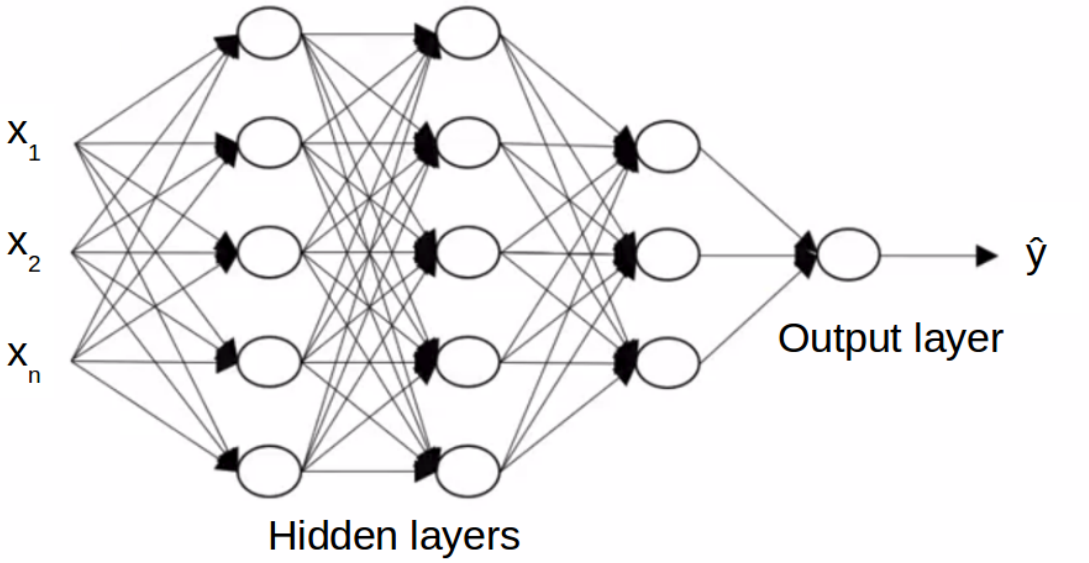

Figure 5.2: A visualization of a neural network with 4 layers, i.e., $L = CAP = 4$. For this network, $n^{[1]} = 5$, $n^{[2]} = 5$, $n^{[3]} = 3$, and $n^{[4]} = 1$.

The basic functions for a vectorized L-layer neural network are the following:

$$A^{[l]} = g^{[l]}(Z^{[l]})\,, \tag{5.17}$$

where $g^{[l]}$ is the activation function for the layer l, such as ReLU or the sigmoid function, and

$$Z^{[l]} = W^{[l]}A^{[l-1]} + B^{[l]}\,. \tag{5.18}$$

The activations $A^{[l]}$ are calculated for each layer in turn, using a for-loop, starting from layer one, i.e., $l = 1$, and then the activations of that layer are fed as input to the next layer, until the output layer $l = L$ is reached. When calculating the value of Z when $l = 1$, i.e., for the first hidden layer, $A^{[0]}$ refers to the input

layer, so that $A^{[0]} = X$. The activations of the output layer are the final results of the algorithm, i.e., $\hat{Y} = A^{[L]}$.

The dimensions of the matrices are as follows. $A^{[0]} = X$ has the same dimensions as in logistic regression, so that it is an $n_x \times m$ matrix, i.e., every column of the matrix corresponds to one training example. More generally, the shape of $A^{[l-1]}$ is the number of neurons in the previous layer times the number of training examples, i.e., $n^{[l-1]} \times m$. In turn, $Z^{[l]}$ and $A^{[l]}$ are $n^{[l]} \times m$ matrices, so that each row contains the activations of one neuron in the given layer to all the training examples. The weights W is an $n^{[l]} \times n^{[l-1]}$ matrix, so that the elements correspond to the connections between the neurons in the previous layer and the current layer. In Figure 5.2, every arrow going to a layer l can be thought of as an element in the matrix $W^{[l]}$. The bias $B^{[l]}$ is an $n^{[l]} \times m$ matrix, whose every column is the same column vector $\mathbf{b}^{[l]}$. Every neuron in layer l has its own bias as an element in the column vector $\mathbf{b}^{[l]}$.

As a sanity check, the sizes of the matrices can be inserted into the Equation (5.18), which results in $(n^{[l]} \times m) = (n^{[l]} \times n^{[l-1]}) \cdot (n^{[l-1]} \times m) + (n^{[l]} \times m)$. The product $(n^{[l]} \times n^{[l-1]}) \cdot (n^{[l-1]} \times m)$ results in a matrix with dimensions $(n^{[l]} \times m)$, so the dimensions of the matrices are compatible.

The same functions for calculating the loss and cost for the results of the output layer can be used as were presented for logistic regression, i.e., the Equations (5.4) and (5.5). The gradient descent also works with the same principle as in Equation (5.6), but now the backward propagation has to be performed through several neural layers, in order to obtain the partial derivatives and update the parameters. The backward propagation starts from the output layer L, goes through all the layers one by one in a for-loop, and finishes in layer $l = 1$. During the backward propagation, each layer receives $\frac{\partial L}{\partial A^{[l]}}$ as an input from the the $(l+1)^{th}$ layer, which is used to calculate $\frac{\partial L}{\partial Z^{[l]}}$, which in turn is used to calculate and return the values for $\frac{\partial L}{\partial A^{[l-1]}}$, $\frac{\partial L}{\partial W^{[l]}}$, and $\frac{\partial L}{\partial B^{[l]}}$ as an output.

For the layer L, i.e., the output layer, the initial backward propagation input $\frac{\partial L}{\partial A^{[l]}}$ is defined in a similar manner as in Equation (5.7), as

$$\frac{\partial L(A^{[L]}, Y)}{\partial A^{[L]}} = -\frac{Y}{A^{[L]}} + \frac{J - Y}{J - A^{[L]}} \tag{5.19}$$

where Y are the known labels for the data, $A^{[L]}$ are the outputs from the output layer of the neural network, and J is a matrix of ones with the same dimension as Y and $A^{[L]}$. The division operations are executed as Hadamard divisions, i.e., element-wise. The derivative $\frac{\partial L}{\partial Z^{[l]}}$ is calculated as

$$\frac{\partial L(A^{[L]}, Y)}{\partial Z^{[l]}} = \frac{\partial L(A^{[L]}, Y)}{\partial A^{[l]}} \circ g^{[l]\prime}(Z^{[l]}) \tag{5.20}$$

where the derivative of g depends on the activation function used. It is $a(1-a)$ for the sigmoid function, and for ReLU it is defined as 1 if $z > 0$, and 0 if $z \leq 0$. The symbol $\circ$ marks the Hadamard product, i.e., the element-wise product. Using the value of $\frac{\partial L}{\partial Z^{[l]}}$, the rest of the derivatives can be calculated:

$$\frac{\partial L(A^{[L]}, Y)}{\partial W^{[l]}} = \frac{1}{m} \frac{\partial L(A^{[L]}, Y)}{\partial Z^{[l]}} A^{[l-1]T} \tag{5.21}$$

$$\frac{\partial L(A^{[L]}, Y)}{\partial B^{[l]}} = \frac{1}{m} \sum_{i=1}^{m} \frac{\partial L(A^{[L]}, Y)}{\partial Z^{[l]}}^{(i)} \tag{5.22}$$

$$\frac{\partial L(A^{[L]}, Y)}{\partial A^{[l-1]}} = W^{[l]T} \frac{\partial L(A^{[L]}, Y)}{\partial Z^{[l]}} \tag{5.23}$$

Furthermore, for $\frac{\partial L(A^{[L]}, Y)}{\partial B^{[l]}}$, the values in each row are summed, and the value of the sum of a given row is placed into every element of that row.

With the derivatives, the values of W and b can be changed accordingly for each layer. After this gradient descent step, the forward propagation step is calculated again, the cost is defined, and the backpropagation is executed again. This iteration is done in a loop until the cost function appears to be optimized.

Some more advanced deep learning algorithms, especially for image recognition, are so-called convolutional neural networks (CNN or ConvNets). The convolutional neural networks have drawn some inspiration from the architecture of

the visual cortex in our brains, and they consist of several different types of layers, such as convolutional layers and pooling layers (LeCun et al., 2015). Another form of deep learning are so-called recurrent neural networks (RNN), i.e., the neurons are not organized strictly in feed-forward layers, but in recurring architectures that makes it possible for the network to learn sequences of inputs, such as words and sentences. An example of RNN is the long short-term memory (LSTM) algorithm (Hochreiter and Schmidhuber, 1997), which is widely used for speech recognition and text generation applications.

5.2.4 Examples of Deep Learning Applications

Some of the most impressive deep learning breakthroughs from the last few years have come from DeepMind, a private artificial intelligence company acquired by Google. DeepMind developed a reinforcement-learning-based AI, which was able to achieve superhuman abilities in many different classic Atari games purely through learning by playing (Mnih et al., 2015). Afterwards, DeepMind focused their attention to the classic game of Go, which is known to be much more complicated than chess. DeepMind's AI, named AlphaGo, was able to defeat a world champion human Go player in the game (Silver et al., 2016). DeepMind reached the milestone of creating a superhuman Go AI a decade earlier than most AI researchers were forecasting. More recently, DeepMind developed a new version of AlphaGo, named AlphaGo Zero, which achieved superhuman playing abilities after only a day of learning, without any specific programmed strategies or knowledge of the game, but only through learning by playing against itself (Silver et al., 2017). Notably, a more generalized version of the AlphaGo Zero AI, named AlphaZero, was able to master all three games of Go, Chess and Shogi, and only after four hours of learning was able to beat all previous chess AIs, which in turn had already been able to beat top human players for a long time (Silver et al., 2017).

In astronomy, deep learning has been used, for example, to find pulsars (Zhu et al., 2014), classify galaxies by morphology (Tuccillo et al., 2016), find exoplanets (Shallue and Vanderburg, 2018), estimate galaxy redshifts (Hoyle, 2016), classify gravitational lenses from image data (Petrillo et al., 2017), detect craters (Cohen et al., 2016; Silburt et al., 2018) and classify fast radio bursts (Connor and van Leeuwen, 2018).

5.3 Implementation for Euclid Data

To test the usefulness of machine learning for detecting asteroid streaks in Euclid data, we programmed the basic algorithms for logistic regression and L-layer neural networks in Python, with the aid of its numerical computing library NumPy. Programs for generating the training data from the existing larger FITS images were also developed. The implemented algorithms were run on a normal Ubuntu laptop computer, using a single CPU.

The main goal of this part of the work was to conduct a feasibility study with regards to applying machine learning methods to simulated Euclid data. The first, most basic goal was to experiment how well a machine learning algorithm could be trained to distinguish between asteroid streak images and non-asteroid images in a simple binary classification task. The second goal was to develop a preliminary method for analyzing the full, 4k×4k-pixel FITS images.

5.3.1 Training data

The training data for the machine learning algorithms was created from the larger FITS files, whose generation is explained in Section 3.2. For the binary classification task, the FITS files were broken into smaller, typically just a few pixels wide images, either clearly containing a part of an asteroid streak (positive training data), or

containing no asteroids at all (negative training data). The positive examples were generated by finding the streaks by their ground-truth coordinates, and generating images of a given size around the middle points of the streaks. The negative examples were generated by taking a random point of the FITS file and checking whether there is a asteroid streak visible in the chosen area or not. If a streak is visible, a new random area is chosen and the procedure is repeated. The negative examples were allowed to contain other types of objects, such as stars, galaxies, and cosmic rays. The ratio between positive and negative examples can be chosen in the program, and it is possible to generate the examples from either one or more of the FITS images at the same time, up to all the 36 FITS files in the chosen magnitude bin. Typically one FITS image contains a few tens of positive examples.

Another type of training data can be generated as well, here called the sliding window training data. It is generated by using a sliding-window algorithm on a larger FITS file, i.e., starting, say, from the upper left corner with a window of given size, checking whether the window in question contains a part of an asteroid streak or not, and according to the result of the check, saving it either as a positive or as a negative training example. After this, the window is moved by a given number of pixels to the right, and the streak checking and training example saving is repeated. When the upper right corner is reached, the window moves back to the left side of the image, a given number of pixels lower than the first row. This procedure is repeated until the whole image is scanned and turned into training examples. Because the whole image contains hugely more negative training examples, i.e., windows with no streaks visible, the amount of negative training examples generated can be limited, and just a random sample of all the negative examples is saved, according to a ratio given by the user.

5.3.2 Algorithms

The machine learning algorithms tested were logistic regression and multi-layer neural network. The implementation of both algorithms follows the equations and architecture laid out in Section 5.2, and we programmed them from the ground up using Python and its NumPy library, which contains highly optimized library functions for matrix operations. The realized logistic regression program consists of approximately 400 lines of code, and the program for deep neural network consists of around 800 lines of code.

The programmed neural network is an L-layer model, so the number of layers and the amount of neurons can be chosen by the user. After some testing, a neural network with the following architecture was chosen for this work: 20 neurons in first hidden layer ($n^{[1]} = 20$), 7 neurons in the second hidden layer ($n^{[2]} = 7$), 5 neurons in third hidden layer ($n^{[3]} = 5$) and 1 output neuron ($n^{[4]} = 1$). Thus, for the chosen network, $CAP = L = 4$. The neurons in the hidden layers have rectified linear units (ReLU) as activation functions, whereas the output neuron activates according to the sigmoid function. The learning rate of the network can be given as a constant, or it can be set to gradually decrease during the learning process, in order to avoid jumping over the minimum in the loss function. During the testing, typically the learning rate α was set to around 0.05 in the beginning, although for some training sets smaller values had to be used in order to keep the value of the cost function converging. The number of stochastic gradient descent iterations was typically from a few hundreds of thousands up to a few tens of millions, depending on the training set.

In order to analyze an entire 4096×4136-pixel image, a sliding window algorithm was used. The basic implementation of the sliding window is the same as that used for generating the sliding window training data (see Section 5.3.1), but now every small sliding-window image is fed to the pre-trained neural network. The

neural network then outputs a probability $\hat{y}$ between 0 and 1 for the image containing a streak. The probability is saved into a heat map, which is then visualized at the end of the sliding window process. The bright areas in the heat map correspond to areas where the neural network gave high probabilities for detecting asteroids, whereas the dark areas correspond to low probabilities.

6. Results

6.1 StreakDet Results

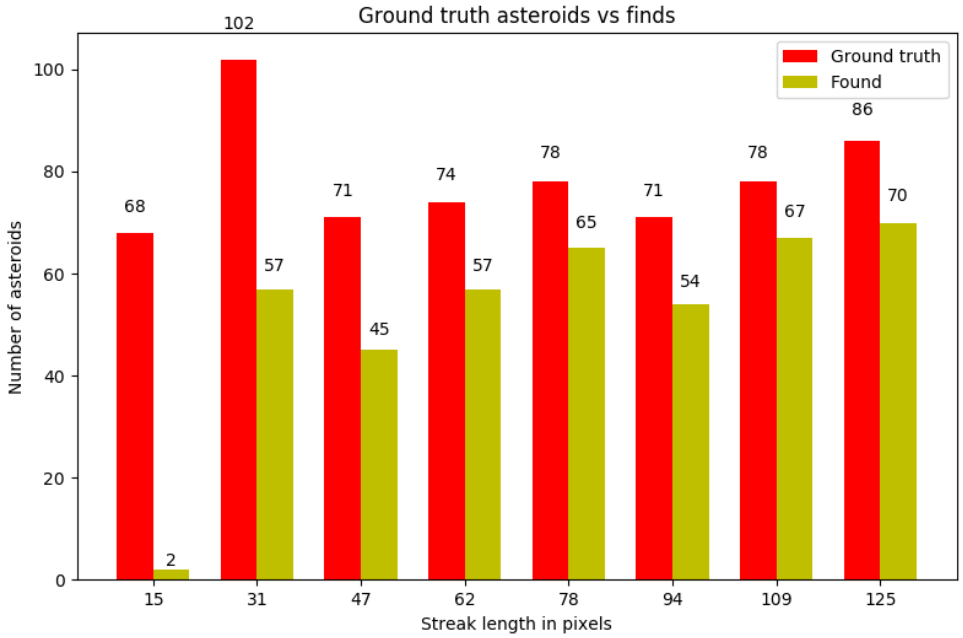

Figure 6.1: Results for SSOs of different lengths after segmentation step in the magnitude range 20–21. The number of ground-truth streaks is shown as red bars, and StreakDet finds as yellow bars. The lengths go from 15 to 125 pixels.

For StreakDet, the fraction of true positives and especially false positives changed radically depending on the settings used. After testing the parameters available for the segmentation phase of the StreakDet pipeline, both in a specific configuration file and StreakDet source code, a few parameters were identified to

slightly increase the fraction of true positives. The segmentation settings are the most important part of tweaking StreakDet, because if a streak is not found already in the segmentation phase, it becomes impossible to find in the later phases either. After testing and optimization, a combination of settings appearing to give good results was chosen, and was used to runs tests for all of the simulated Euclid data. The results after segmentation for the brightest streaks, in the magnitude range 20–21, are shown in Figure 6.1.

After the segmentation step, there was still a huge number of false positives in the pipeline, typically around 100,000. Most false positives are discarded during the later stages of the StreakDet pipeline, but naturally a few of the true positives are also lost in the process. The final StreakDet results, after all the processing stages, are shown in Figure 6.2.

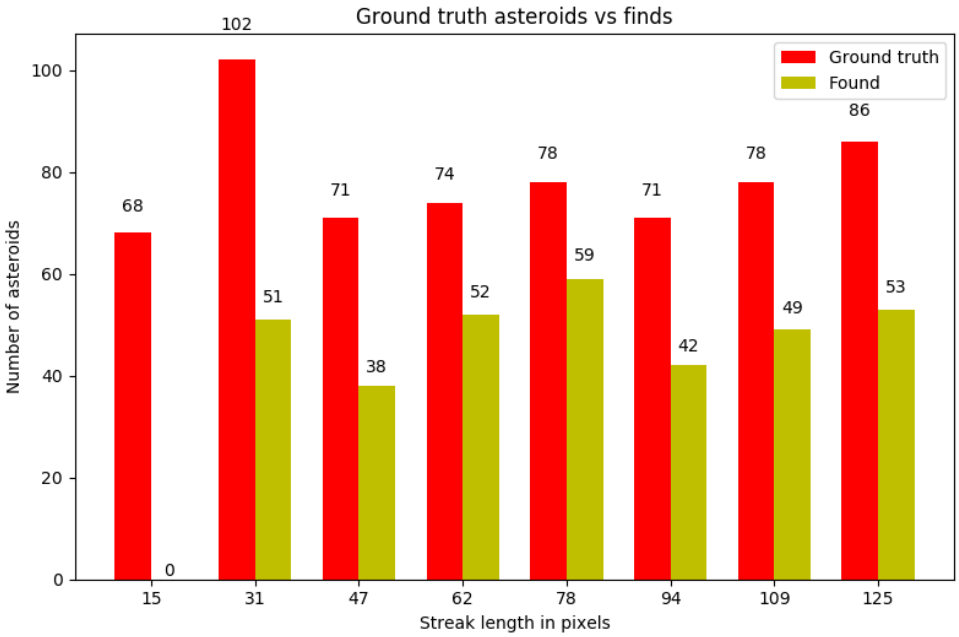

Figure 6.2: Final results for SSOs of different lengths in magnitude range 20–21.

After segmentation, 66.4% (417 out of 628) of all streaks in magnitude range 20–21 were found. Only very few streaks with lengths of 15 pixels were found, which

corresponds to an SSO velocity of 10 "/h, but for lengths of 30 pixels (20 "/h) and higher the percentage was better.

In the final results, 54.8% (344 out of 628) of all streaks in magnitude range 20–21 were found. No 15-pixel streaks were found, but for all longer streaks the percentage was consistently above 50%. In the final results there were 102 false positives. The false positives were generally from either galaxies, cosmic rays or "donuts" (ghost starlight of bright stars), or some combination thereof.

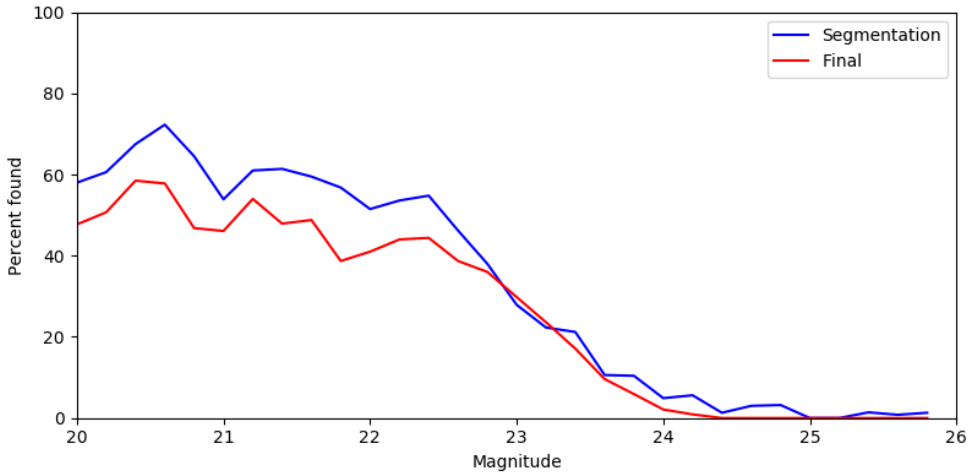

Figure 6.3: StreakDet finding percentage for SSOs of different magnitudes. The blue line shows findings in segmentation phase whereas the red line marks the final results.

The finding percentage was fairly consistent for magnitudes below 22.5. For dimmer streaks the percentage started to decrease, falling close to zero after magnitude 24, as can be seen in Figure 6.3. At around magnitude 23 there appears to be slightly more finds in the final results than in the earlier segmentation phase, which should be impossible. This oddity is explained by the fact that the coordinates and angles in StreakDet data in the segmentation phase are not very accurate, which causes some of the true positives in the segmentation phase to be classified as false positives. However, that only means that the StreakDet finding percentage for segmentation results was in reality slightly higher than the plots show. The

final StreakDet results have much more accurate coordinates and angles, and the classification error vanishes there.

The angles of final StreakDet finds were typically very accurate when compared to ground truth. For the brightest streaks, in magnitude range 20–22, the average angle error was only around 0.02 degrees. For dimmer streaks the average error increased a bit, but even for the dimmest found streaks, in magnitude range 24–25, the average error was still only 0.16 degrees. The lengths of the StreakDet finds had less accuracy, and were often shorter than the corresponding ground-truth streaks. For long streaks, StreakDet sometimes found two shorter line segments of the ground truth streak instead of the whole streak.

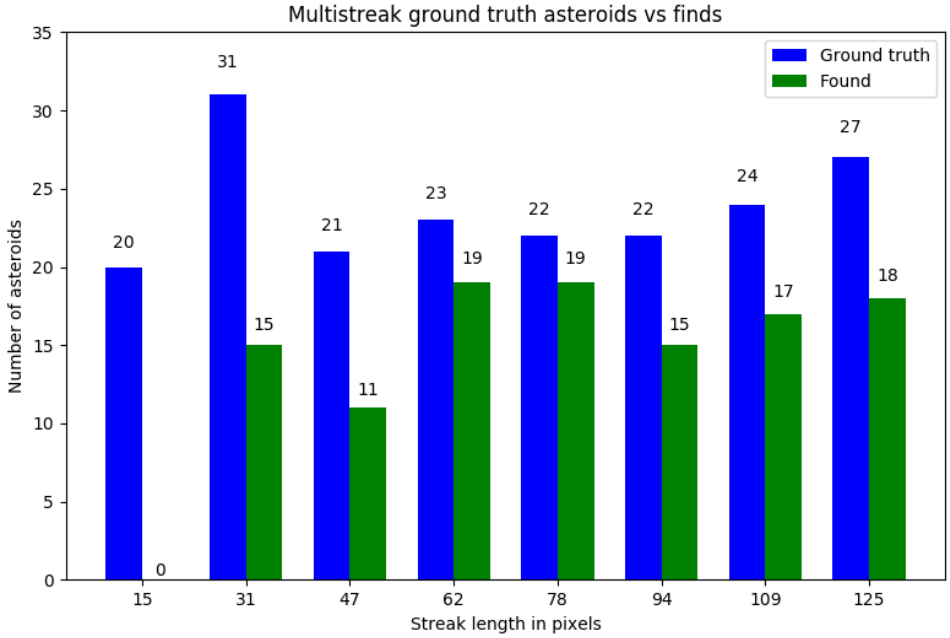

Figure 6.4: Results of the multistreak analysis for magnitudes in range 20–21. The blue bars show the number of ground truth multistreaks, while the green bars show the number of StreakDet multistreaks.

Multistreak analysis managed to discard virtually all of the false positives, while maintaining most of the true positives. Ground-truth objects that appeared

only in one dither were dropped, as were StreakDet finds that did not fall on the same line with other StreakDet finds. Figure 6.4 shows the results for multistreaks of different lengths, from the final StreakDet results, for magnitude range 20–21. After this particular multistreak analysis, there were zero false positives left. For magnitude range 20–21, the find percentage was 60.0% after multistreak analysis, compared to the 54.8% for single-streak analysis. This increase is due to the fact that StreakDet only has to find two of the single streaks from the multistreak. In other words, if the ground-truth multistreak consists of four single streaks, and StreakDet finds only two of them, that is still a hit.

The comprehensive StreakDet results can be found in Appendix A, which includes results for all six magnitude ranges after segmentation, after the whole pipeline, and after multistreak analysis.

6.2 Deep Learning Results

In binary classification, logistic regression and neural networks gave best results for small images. Images with sizes up to 10×10 pixels gave relatively good results, and the smallest, only 2 pixels wide images offered the best classification accuracies (see Table 6.1). For larger images, for example 130x130, where an entire streak is visible, a classifier based on random guesses would have produced almost the same accuracy, 50%, as the neural network. The main reason is that the amount of training data was not enough to teach the neural network what exactly it should look for — an image with a size of 130x130 contains 16900 pixels and therefore requires a lot of training data.

For small images with a width of a few pixels, a basic neural network learned to classify between streak and non-streak images with up to 98% accuracy, for the brightest asteroid streaks in the 20–21 magnitude bin. The classification accuracy started to decrease when moving to fainter streaks (see Table 6.2).

Table 6.1: Classification accuracies for images of different sizes for the logistic regression and neural network algorithms. The accuracy refers to the percentage of correctly classified images. In the training, cross-validation and test set there was an equal number of positive (containing a part of an asteroid streak) and negative (not containing asteroids) training examples in magnitude range 20–21. For every image size, both the training and test set contained approximately 150 images.

Image size (pixels)	Logistic regression accuracy	Neural network accuracy
50×50	58.33%	60.81%
30×30	66.00%	68.89%
20×20	76.33%	80.67%
10×10	84.55%	90.00%
8×8	87.16%	91.39%
6×6	91.20%	94.28%
4×4	90.49%	96.46%
2×2	92.21%	98.48%

Visual testing with the sliding window approach showed that when analyzing a complete, larger FITS file, a large number of false positives were found, but they were typically spot-like features in the generated heat map. The neural network activated to some extent on pixels with cosmic rays, so-called donuts, and some galaxies, but typically the correct streaks were found most clearly, and they stand out from the heat map. In the basic classification training set there was typically the same number of positive and negative training examples. In the sliding window detection, there is a vastly larger number of negative areas (areas without asteroids) than positive areas, so the sliding window neural network gave better results when it was trained with data containing more negative examples than positive ones.

The tested machine learning algorithms appear to have learned to detect as-

Table 6.2: Classification accuracies for 2×2-pixel images containing parts of asteroid streaks from certain magnitude ranges, for both the logistic regression and neural network algorithms. In the training, cross-validation, and test sets there was an equal number of positive and negative training examples. For each magnitude range, each of the sets contained streaks only from the designated magnitude range. An exception is the range from 25 to 26, for which algorithms trained with examples from magnitude range 24–25 gave the best results.

Magnitude	Logistic regression accuracy	Neural network accuracy
20–21	92.21%	98.48%
21–22	89.67%	95.31%
22–23	85.97%	91.26%
23–24	78.39%	83.26%
24–25	65.46%	67.1%
25–26	58.60%	58.91%

teroids mostly by the magnitudes of the pixels, because that is the most obvious feature of the streaks to learn. When comparing the images generated by the sliding window neural network and images generated by a simple magnitude filter, the asteroids stand out more clearly in the heat map generated by the neural network. This means that the algorithm learned to detect other features of the streaks as well, in addition to magnitude, even from the small 2×2-pixel training images. This observation was enforced by the fact the a neural network taught only with dim streaks was able to classify also brighter streaks with a good accuracy, and sometimes with even higher accuracy than a network trained with the bright streaks. A network taught only with bright streaks, on the other hand, typically had problems classifying dimmer streaks. A peculiar case was the magnitude range from 25 to 26, for which the streaks are already practically indistinguishable from background noise. The algorithms trained with examples from this range were barely able to

reach accuracies any higher than the baseline of 50%. However, when testing the dimmest examples with algorithms trained with magnitude 24–25 data, the accuracies jumped closer to 60%.

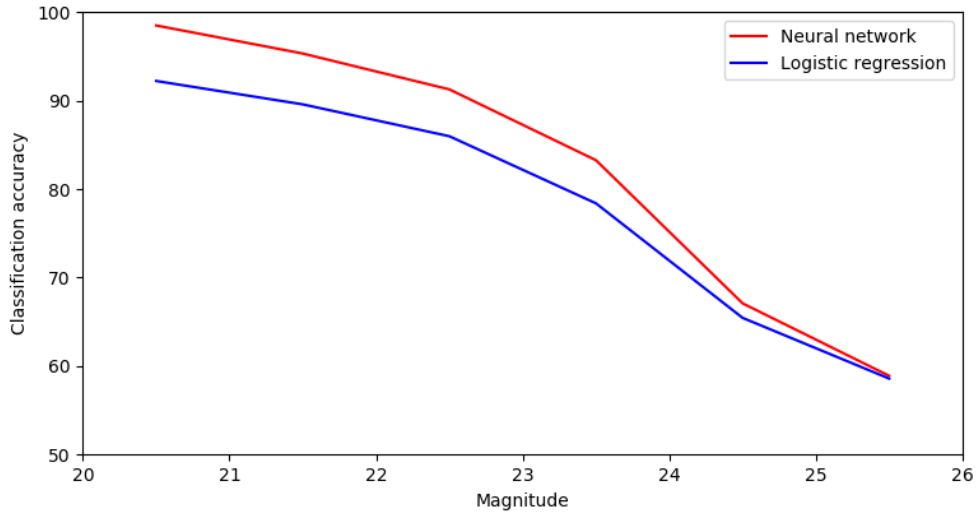

Figure 6.5: The plot shows the same binary classification accuracies as are presented in Table 6.2. Y-axis is limited to 50%, as that is the baseline accuracy that could be achieved with random guessing.

The time it took to train the logistic regression or neural network algorithms was typically from a few minutes to a few tens of minutes for one training set, after sensible hyperparameters were first found. The classification of the cross-validation set or test set was executed practically instantaneously. The sliding window algorithm took 2–3 minutes to analyze a full 4k×4k-pixel image.

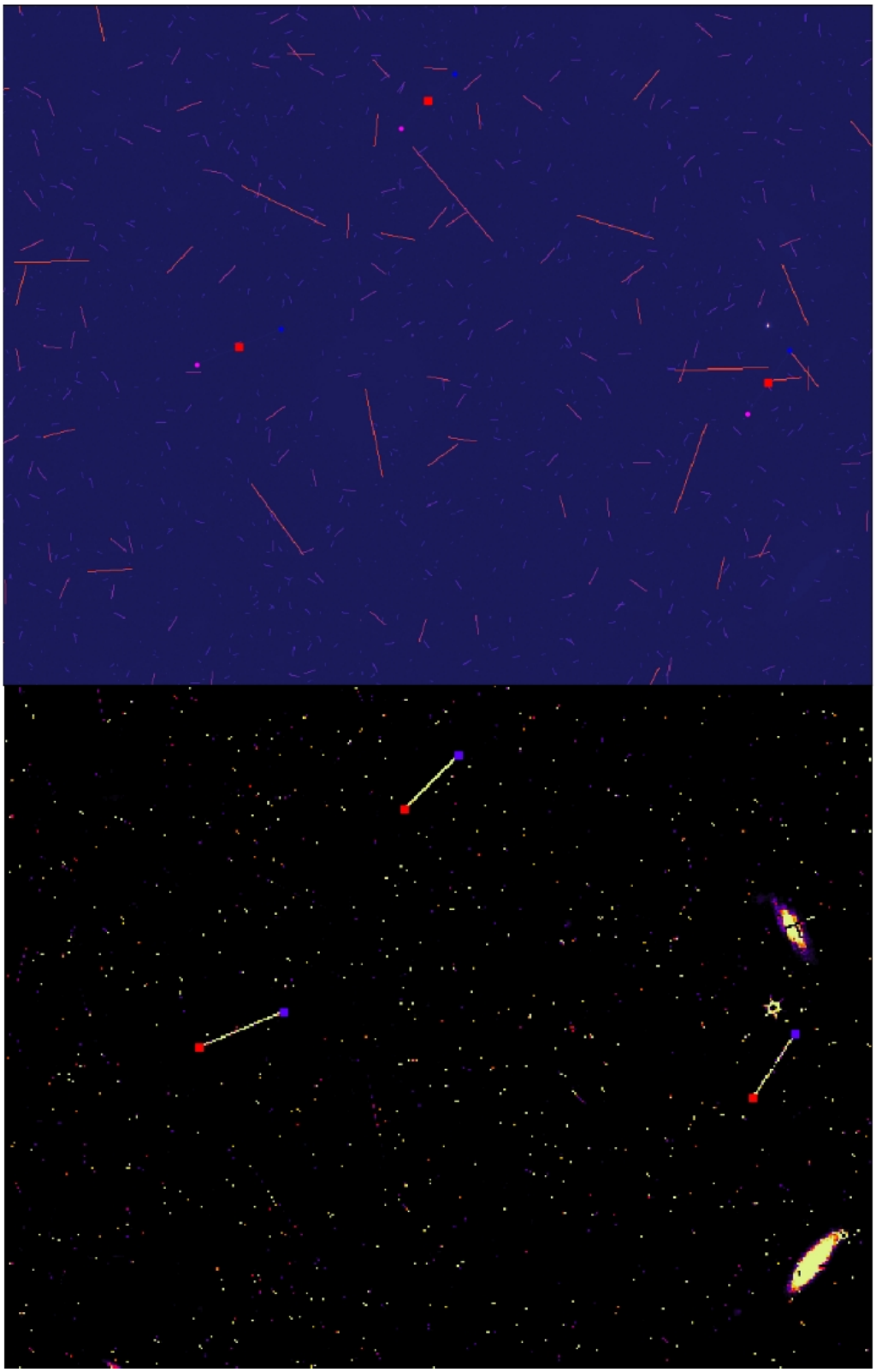

Figure 6.6: The upper image shows a 900 pixels (90 arcseconds) wide and 700 pixels (70 arcseconds) high area of an original FITS image, containing the three marked asteroid streaks in the magnitude range of 20–21. The lower image shows a heat map generated for the same area by the sliding window neural network, in which the asteroids are clearly identifiable. For further clarity, the positions of asteroids in the heat map are also marked with the red and blue squares.

7. Discussion

The StreakDet results were relatively good, with around 60% detection rate for brightest streaks, when utilizing the multistreak algorithm, and slightly above 50%, when utilizing the standard single-streak approach. Nevertheless, a higher detection rate would be preferred. StreakDet had problems when detecting short streaks, i.e., streaks with lengths of around 15 pixels and shorter. The streaks that were not found were typically lost already in the segmentation phase. The number of false positives depended a lot on the settings used, but the developed multistreak analysis was able to get rid of virtually all of the false positives. For this reason, it would be preferred that the segmentation algorithm could be improved to find a larger portion of the real streaks, even if that would cause an increase in the number of false positives at the single-streak-analysis stage, since virtually all of the false positives can be dropped at the multistreak-analysis stage.

Later phases of the StreakDet pipeline seemed to work well, even though not much emphasis was put on optimizing their parameters yet. The angles of the final streaks matched the ground truth exceptionally well. Also, the coordinates of the final found streaks matched with the ground truth reasonably well, except in the cases when StreakDet found only a shorter part of a long streak.

The machine learning algorithms used in this work, logistic regression and multi-layer neural network, were relatively basic ones and at this stage were tested more as a proof-of-concept rather than as an attempt to create a ready-to-deploy

streak detection software. The results achieved with these algorithms were encouraging, reaching up to 98% accuracy in binary classification.

When testing the larger images, i.e, wider than 10 pixels, the machine learning algorithms were clearly overfitting to the training set. The algorithms had close to 100% accuracies when tested on the training set, but several tens of percentage points lower accuracies when tested on the cross-validation sets and test sets. In some sense, overfitting to the training data can be thought of as simply memorizing the training set, and then having problems classifying images that have not been seen before. The reason for overfitting, in this case, was the low amount of training data compared to the high dimensionality of the larger images. For small images, the amount of pixels, and thus feature vector dimensions, is much smaller when compared to the number of training examples, which limits overfitting.

There is a lot of room for improvement in the detection ability of the neural network. Furthermore, a method for returning the actual coordinates of the asteroids from the full FITS images could be developed. Using a convolutional neural network would very likely radically improve the classification accuracies, and perhaps could be trained to directly output the streak coordinates, instead of just the classification results $\hat{y}$. If it turns out difficult for a single deep learning algorithm to learn to output the streak coordinates directly from the raw data, a program consisting of two separate neural networks could work. The first deep learning model could produce a similar sliding window heat map as was already developed for this work, and then another deep learning model could be trained, on the basis of the generated heat map data, to return the coordinates of the asteroid features from this simpler visual data.

Although the training stages of the machine learning algorithms were optimized in the sense that they were vectorized, no other performance-enhancing optimizations were carried out. The learning stage could be further accelerated with

utilizing a GPU instead of the single CPU that was used now. This would make it possible to have much larger training sets, which in turn could help increase the classification accuracies, especially for larger images. The sliding-window algorithm could likely be vectorized as well, decreasing the time it takes to search for asteroids in a large FITS image from the current 2–3 minutes.

For this work, we programmed the machine learning algorithms from the ground up, without exploiting any ready-made machine learning libraries. The main reason was to get acquainted with the algorithms and their mathematics. Henceforth, using highly optimized machine learning libraries, such as Tensorflow or PyTorch, will make more sense, as it will be much easier to build complex and powerful deep learning models with them.

The simulated Euclid images should correspond quite well to the upcoming real Euclid images, but naturally the resemblance cannot reach 100%. Therefore, if deep learning will be used to find asteroids in the real Euclid data, a neural network trained only with simulated data will probably not work optimally from the get-go. Instead, the algorithm will probably have to be trained with at least some examples of real asteroid streaks and non-asteroid images picked from the real Euclid data. However, if the difference between the simulated and real data turns out to be small, it could be possible to train the model first with the plentiful simulated data, and continue training the pre-trained neural network with real Euclid data, as it comes in. This procedure could limit the amount of real training data needed. Furthermore, even without pre-training, this work has shown that a deep learning model can learn to detect asteroids streaks with a relatively small amount of training data, typically around 150 training images.

Going forward, one potential approach could be combining the best of both tested approaches. Neural networks seem well-suited for finding the rough coordinates of the asteroid streaks from Euclid data. StreakDet had problems with exactly

that initial process, but worked well for getting accurate angles and coordinates of the streaks, once they were present in the pipeline of the software. Therefore, neural networks could be utilized in the segmentation phase of StreakDet to find the crude coordinates of potential streaks, and then the later stages of StreakDet could be used to extract the exact coordinates and angles of the streaks.

8. Conclusions

One of the limiting factors restraining our understanding of the geophysical and geological properties of asteroid populations is the lack of proper spectral data from the objects. The upcoming Euclid mission will ease this problem by offering a large amount of visual and near-infrared spectral data of a large number of asteroids. The asteroids will appear as faint streaks in the data, and there are no ready-to-deploy software to detect them from the vast amount of Euclid images. We tested two potential methods for detecting the asteroids, both of which showed promise.

StreakDet found approximately 60% of SSOs in the simulated Euclid data, when the lengths of the streaks were above 15 pixels, the magnitudes of the objects were 22.5 or brighter, and when the multistreak algorithm was used. At magnitude 22.5 the finding percentage started to clearly decrease, and reached zero after magnitude 24. The multistreak analysis run on the final StreakDet results worked well in retaining most of the correct finds, and even slightly increasing the total detection percentage, while removing almost all of the false positives. The standard single-streak analysis had detection accuracies of a few percentage points lower across the board, and typically had a fairly notable fraction of false positives.

As a proof-of-concept, applying artificial neural networks to find streaks in the Euclid data worked well. A 4-layer neural network was able to learn to differentiate between small images containing asteroid streaks and images not containing streaks with up to 98% accuracy. The preliminary sliding window approach for analyzing

larger images was able to visualize the rough positions of asteroids well, although a fair number of noise and false positives was still present in the final heat map generated by the sliding-window neural network.

Going forward, the areas with most potential for improving the overall detection ability of asteroids in Euclid data are improving the segmentation phase of StreakDet, or developing a more advanced deep learning model, such as a convolutional neural network, that is capable of directly returning the coordinates of the asteroids in the images. One possible option is to combine these two approaches.

Bibliography

A'Hearn, M. F., Belton, M. J. S., Delamere, W. A., Kissel, J., Klaasen, K. P., McFadden, L. A., Meech, K. J., Melosh, H. J., Schultz, P. H., Sunshine, J. M., Thomas, P. C., Veverka, J., Yeomans, D. K., Baca, M. W., Busko, I., Crockett, C. J., Collins, S. M., Desnoyer, M., Eberhardy, C. A., Ernst, C. M., Farnham, T. L., Feaga, L., Groussin, O., Hampton, D., Ipatov, S. I., Li, J.-Y., Lindler, D., Lisse, C. M., Mastrodemos, N., Owen, W. M., Richardson, J. E., Wellnitz, D. D., and White, R. L. (2005). Deep Impact: Excavating comet Tempel 1. *Science*, 310(5746):258–264.

Alcock, C., Fristrom, C. C., and Siegelman, R. (1986). On the number of comets around other single stars. *Astrophysical Journal*, 302:462–476.

Alvarez, L. W., Alvarez, W., Asaro, F., and Michel, H. V. (1980). Extraterrestrial cause for the Cretaceous-Tertiary extinction. *Science*, 208(4448):1095–1108.

Amendola, L., Appleby, S., Avgoustidis, A., Bacon, D., Baker, T., Baldi, M., Bartolo, N., Blanchard, A., Bonvin, C., Borgani, S., Branchini, E., Burrage, C., Camera, S., Carbone, C., Casarini, L., Cropper, M., de Rham, C., Dietrich, J. P., Di Porto, C., Durrer, R., Ealet, A., Ferreira, P. G., Finelli, F., Garcia-Bellido, J., Giannantonio, T., Guzzo, L., Heavens, A., Heisenberg, L., Heymans, C., Hoekstra, H., Hollenstein, L., Holmes, R., Horst, O., Hwang, Z., Jahnke, K., Kitching, T. D., Koivisto, T., Kunz, M., La Vacca, G., Linder, E., March, M., Marra, V.,

Martins, C., Majerotto, E., Markovic, D., Marsh, D., Marulli, F., Massey, R., Mellier, Y., Montanari, F., Mota, D. F., Nunes, N. J., Percival, W., Pettorino, V., Porciani, C., Quercellini, C., Read, J., Rinaldi, M., Sapone, D., Sawicki, I., Scaramella, R., Skordis, C., Simpson, F., Taylor, A., Thomas, S., Trotta, R., Verde, L., Vernizzi, F., Vollmer, A., Wang, Y., Weller, J., and Zlosnik, T. (2016). Cosmology and fundamental physics with the Euclid satellite. *ArXiv e-prints*.

Binzel, R. P. (2000). The Torino impact hazard scale. *Planetary and Space Science*, 48(4):297 – 303.

Bostrom, N. (2013). Existential risk prevention as global priority. *Global Policy*, 4(1):15–31.

Brown, P., Spalding, R. E., ReVelle, D. O., Tagliaferri, E., and Worden, S. P. (2002). The flux of small near-Earth objects colliding with the Earth. *Nature*, 420:294–296.

Bus, S. J. (1999). *Compositional structure in the asteroid belt: Results of a spectroscopic survey*. PhD thesis, Massachusetts institute of technology.

Carry, B. (2018). Solar system science with ESA Euclid. *Astronomy and Astrophysics*, 609:A113.

Chesley, S. R., Chodas, P. W., Milani, A., Valsecchi, G. B., and Yeomans, D. K. (2002). Quantifying the risk posed by potential Earth impacts. *Icarus*, 159(2):423 – 432.

Cohen, J. P., Lo, H. Z., Lu, T., and Ding, W. (2016). Crater detection via convolutional neural networks. In *Lunar and Planetary Science Conference*, volume 47 of *Lunar and Planetary Science Conference*, page 1143.

Connelly, J. N., Bizzarro, M., Krot, A. N., Nordlund, Å., Wielandt, D., and Ivanova,

M. A. (2012). The absolute chronology and thermal processing of solids in the solar protoplanetary disk. *Science*, 338:651.

Connor, L. and van Leeuwen, J. (2018). Applying deep learning to fast radio burst classification. *ArXiv e-prints*.

Cox, D. R. (1958). The regression analysis of binary sequences. *Journal of the Royal Statistical Society. Series B (Methodological)*, 20(2):215–242.

DeMeo, F. E., Alexander, C. M. O., Walsh, K. J., Chapman, C. R., and Binzel, R. P. (2015). The compositional structure of the asteroid belt. In Michel, P., DeMeo, F. E., and Bottke, W. F., editors, *Asteroids IV*, pages 13–41.

DeMeo, F. E., Binzel, R. P., Slivan, S. M., and Bus, S. J. (2009). An extension of the Bus asteroid taxonomy into the near-infrared. *Icarus*, 202(1):160 – 180.

Glorot, X., Bordes, A., and Bengio, Y. (2011). Deep sparse rectifier neural networks. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pages 315–323.

Gomes, R., Levison, H. F., Tsiganis, K., and Morbidelli, A. (2005). Origin of the cataclysmic Late Heavy Bombardment period of the terrestrial planets. *Nature*, 435:466–469.

Granvik, M., Morbidelli, A., Jedicke, R., Bolin, B., Bottke, W. F., Beshore, E., Vokrouhlický, D., Delbò, M., and Michel, P. (2016). Super-catastrophic disruption of asteroids at small perihelion distances. *Nature*, 530:303–306.

Harris, A. W., Boslough, M., Chapman, C. R., Drube, L., and Michel, P. (2015). Asteroid impacts and modern civilization: Can we prevent a catastrophe? In Michel, P., DeMeo, F. E., and Bottke, W. F., editors, *Asteroids IV*, pages 835–854.

Hildebrand, A. R., Penfield, G. T., Kring, D. A., Pilkington, M., Camargo Z., A., Jacobsen, S. B., and Boynton, W. V. (1991). Chicxulub crater: A possible Cretaceous/Tertiary boundary impact crater on the Yucatan Peninsula, Mexico. *Geology*, 19(9):867.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Hoyle, B. (2016). Measuring photometric redshifts using galaxy images and deep neural networks. *Astronomy and Computing*, 16:34 – 40.

Ivezić, Ž., Tabachnik, S., Rafikov, R., Lupton, R. H., Quinn, T., Hammergren, M., Eyer, L., Chu, J., Armstrong, J. C., Fan, X., Finlator, K., Geballe, T. R., Gunn, J. E., Hennessy, G. S., Knapp, G. R., Leggett, S. K., Munn, J. A., Pier, J. R., Rockosi, C. M., Schneider, D. P., Strauss, M. A., Yanny, B., Brinkmann, J., Csabai, I., Hindsley, R. B., Kent, S., Lamb, D. Q., Margon, B., McKay, T. A., Smith, J. A., Waddel, P., York, D. G., and SDSS Collaboration (2001). Solar System objects observed in the Sloan Digital Sky Survey commissioning data. *The Astronomical Journal*, 122:2749–2784.

Jewitt, D., Hsieh, H., and Agarwal, J. (2015). The active asteroids. In Michel, P., DeMeo, F. E., and Bottke, W. F., editors, *Asteroids IV*, pages 221–241.

Joachimi, B. (2016). Euclid - an ESA medium class mission. In Skillen, I., Balcells, M., and Trager, S., editors, *Multi-Object Spectroscopy in the Next Decade: Big Questions, Large Surveys, and Wide Fields*, volume 507 of *Astronomical Society of the Pacific Conference Series*, page 401.

Jones, R. L., Chesley, S. R., Connolly, A. J., Harris, A. W., Ivezic, Z., Knezevic, Z., Kubica, J., Milani, A., and Trilling, D. E. (2009). Solar System science with LSST. *Earth Moon and Planets*, 105:101–105.

Jura, M. (2008). Pollution of single white dwarfs by accretion of many small asteroids. *The Astronomical Journal*, 135:1785–1792.

Krasinsky, G. A., Pitjeva, E. V., Vasilyev, M. V., and Yagudina, E. I. (2002). Hidden Mass in the Asteroid Belt. *Icarus*, 158:98–105.

Lawler, S. M. and Gladman, B. (2012). Debris disks in Kepler exoplanet systems. *The Astrophysical Journal*, 752(1):53.

LeCun, Y., Bengio, Y., and Hinton, G. E. (2015). Deep learning. *Nature*, 521(7553):436–444.

Lewis, J. (2015). *Asteroid mining 101: Wealth for the new space economy:*. Deep Space Industries Incorporated.

Mainzer, A., Bauer, J., Grav, T., Masiero, J., Cutri, R. M., Dailey, J., Eisenhardt, P., McMillan, R. S., Wright, E., Walker, R., Jedicke, R., Spahr, T., Tholen, D., Alles, R., Beck, R., Brandenburg, H., Conrow, T., Evans, T., Fowler, J., Jarrett, T., Marsh, K., Masci, F., McCallon, H., Wheelock, S., Wittman, M., Wyatt, P., DeBaun, E., Elliott, G., Elsbury, D., Gautier, IV, T., Gomillion, S., Leisawitz, D., Maleszewski, C., Micheli, M., and Wilkins, A. (2011). Preliminary results from NEOWISE: An enhancement to the Wide-field Infrared Survey Explorer for Solar System science. *The Astrophysical Journal*, 731:53.

McCarthy, J., Minsky, M. L., Rochester, N., and Shannon, C. E. (1955). A proposal for the Dartmouth Summer Research Project on Artificial Intelligence: August 31, 1955.

McCulloch, W. S. and Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133.

Meech, K. J., Weryk, R., Micheli, M., Kleyna, J. T., Hainaut, O. R., Jedicke, R., Wainscoat, R. J., Chambers, K. C., Keane, J. V., Petric, A., Denneau, L.,

Magnier, E., Berger, T., Huber, M. E., Flewelling, H., Waters, C., Schunova-Lilly, E., and Chastel, S. (2017). A brief visit from a red and extremely elongated interstellar asteroid. *Nature*, 552:378–381.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., and Hassabis, D. (2015). Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533.

Morbidelli, A., Brasser, R., Gomes, R., Levison, H. F., and Tsiganis, K. (2010). Evidence from the Asteroid Belt for a violent past evolution of Jupiter's orbit. *The Astronomical Journal*, 140:1391–1401.

Morbidelli, A., Levison, H. F., Tsiganis, K., and Gomes, R. (2005). Chaotic capture of Jupiter's Trojan asteroids in the early Solar System. *Nature*, 435:462–465.

Niemi, S.-M. (2015). Euclid Visible InStrument (VIS) Python Package (VIS-PP).

Petrillo, C. E., Tortora, C., Chatterjee, S., Vernardos, G., Koopmans, L. V. E., Verdoes Kleijn, G., Napolitano, N. R., Covone, G., Schneider, P., Grado, A., and McFarland, J. (2017). Finding strong gravitational lenses in the Kilo degree survey with convolutional neural networks. *Monthly Notices of the Royal Astronomical Society*, 472(1):1129–1150.

Popova, O. P., Jenniskens, P., Emel'yanenko, V., Kartashova, A., Biryukov, E., Khaibrakhmanov, S., Shuvalov, V., Rybnov, Y., Dudorov, A., Grokhovsky, V. I., Badyukov, D. D., Yin, Q.-Z., Gural, P. S., Albers, J., Granvik, M., Evers, L. G., Kuiper, J., Kharlamov, V., Solovyov, A., Rusakov, Y. S., Korotkiy, S., Serdyuk, I., Korochantsev, A. V., Larionov, M. Y., Glazachev, D., Mayer, A. E., Gisler, G., Gladkovsky, S. V., Wimpenny, J., Sanborn, M. E., Yamakawa, A., Verosub,

K. L., Rowland, D. J., Roeske, S., Botto, N. W., Friedrich, J. M., Zolensky, M. E., Le, L., Ross, D., Ziegler, K., Nakamura, T., Ahn, I., Lee, J. I., Zhou, Q., Li, X.-H., Li, Q.-L., Liu, Y., Tang, G.-Q., Hiroi, T., Sears, D., Weinstein, I. A., Vokhmintsev, A. S., Ishchenko, A. V., Schmitt-Kopplin, P., Hertkorn, N., Nagao, K., Haba, M. K., Komatsu, M., Mikouchi, T., and (2013). Chelyabinsk airburst, damage assessment, meteorite recovery, and characterization. *Science*, 342(6162):1069–1073.

Reddy, V., Dunn, T. L., Thomas, C. A., Moskovitz, N. A., and Burbine, T. H. (2015). Mineralogy and surface composition of asteroids. In Michel, P., DeMeo, F. E., and Bottke, W. F., editors, *Asteroids IV*, pages 43–63.

Russell, S. and Norvig, P. (2009). *Artificial intelligence: A modern approach*. Prentice Hall Press, Upper Saddle River, NJ, USA, 3rd edition.

Shallue, C. J. and Vanderburg, A. (2018). Identifying exoplanets with deep learning: A five-planet resonant chain around Kepler-80 and an eighth planet around Kepler-90. *The Astronomical Journal*, 155(2):94.

Silburt, A., Ali-Dib, M., Zhu, C., Jackson, A., Valencia, D., Kissin, Y., Tamayo, D., and Menou, K. (2018). Lunar crater identification via deep learning. *ArXiv e-prints*.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., van den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D., Nham, J., Kalchbrenner, N., Sutskever, I., Lillicrap, T., Leach, M., Kavukcuoglu, K., Graepel, T., and Hassabis, D. (2016). Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587):484–489.

Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., Lanctot, M., Sifre, L., Kumaran, D., Graepel, T., Lillicrap, T., Simonyan, K., and Hassabis,

D. (2017). Mastering Chess and Shogi by self-play with a general reinforcement learning algorithm. *ArXiv e-prints*.

Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., Chen, Y., Lillicrap, T., Hui, F., Sifre, L., Driessche, G. v. d., Graepel, T., and Hassabis, D. (2017). Mastering the game of Go without human knowledge. *Nature*, 550(7676):354–359.

Sunshine, J. M., Pieters, C. M., and Pratt, S. F. (1990). Deconvolution of mineral absorption bands: An improved approach. *Journal of Geophysical Research: Solid Earth*, 95(B5):6955–6966.

Tholen, D. J. (1984). *Asteroid taxonomy from cluster analysis of photometry*. PhD thesis, The University of Arizona.

Tsiganis, K., Gomes, R., Morbidelli, A., and Levison, H. F. (2005). Origin of the orbital architecture of the giant planets of the Solar System. *Nature*, 435:459–461.

Tuccillo, D., Huertas-Company, M., Decencière, E., and Velasco-Forero, S. (2016). Deep learning for studies of galaxy morphology. *Proceedings of the International Astronomical Union*, 12(S325):191–196.

Turing, A. M. (1950). I.—Computing machinery and intelligence. *Mind*, LIX(236):433–460.

Virtanen, J., Poikonen, J., Säntti, T., Komulainen, T., Torppa, J., Granvik, M., Muinonen, K., Pentikäinen, H., Martikainen, J., Näränen, J., Lehti, J., and Flohrer, T. (2016). Streak detection and analysis pipeline for space-debris optical images. *Advances in Space Research*, 57(8):1607 – 1623.

Väisälä, Y. (1939). Über die Planetenbeobachtungen an der Sternwarte der Universität Turku. *Astronomische Nachrichten*, 268(1):7–10.

Walsh, K. J., Morbidelli, A., Raymond, S. N., O'Brien, D. P., and Mandell, A. M. (2011). A low mass for Mars from Jupiter's early gas-driven migration. *Nature*, 475:206–209.

Weidenschilling, S. J. (1977). The distribution of mass in the planetary system and solar nebula. *Astrophysics and Space Science*, 51:153–158.

Zhu, W. W., Berndsen, A., Madsen, E. C., Tan, M., Stairs, I. H., Brazier, A., Lazarus, P., Lynch, R., Scholz, P., Stovall, K., Ransom, S. M., Banaszak, S., Biwer, C. M., Cohen, S., Dartez, L. P., Flanigan, J., Lunsford, G., Martinez, J. G., Mata, A., Rohr, M., Walker, A., Allen, B., Bhat, N. D. R., Bogdanov, S., Camilo, F., Chatterjee, S., Cordes, J. M., Crawford, F., Deneva, J. S., Desvignes, G., Ferdman, R. D., Freire, P. C. C., Hessels, J. W. T., Jenet, F. A., Kaplan, D. L., Kaspi, V. M., Knispel, B., Lee, K. J., van Leeuwen, J., Lyne, A. G., McLaughlin, M. A., Siemens, X., Spitler, L. G., and Venkataraman, A. (2014). Searching for pulsars using image pattern recognition. *The Astrophysical Journal*, 781(2):117.

A. StreakDet results for all tested magnitudes

We ran the StreakDet tests on six separate sets of simulated Euclid data, each containing asteroids in designated magnitude ranges. The first set contained asteroids with magnitudes between 20 and 21, the second set between 21 and 22, and so on, until the set with the faintest streaks with magnitudes between 25 and 26. The following bar charts visualize the StreakDet results for all the sets, both after the segmentation step, and after the whole StreakDet pipeline. The results after multistreak analysis are also presented for each magnitude range.

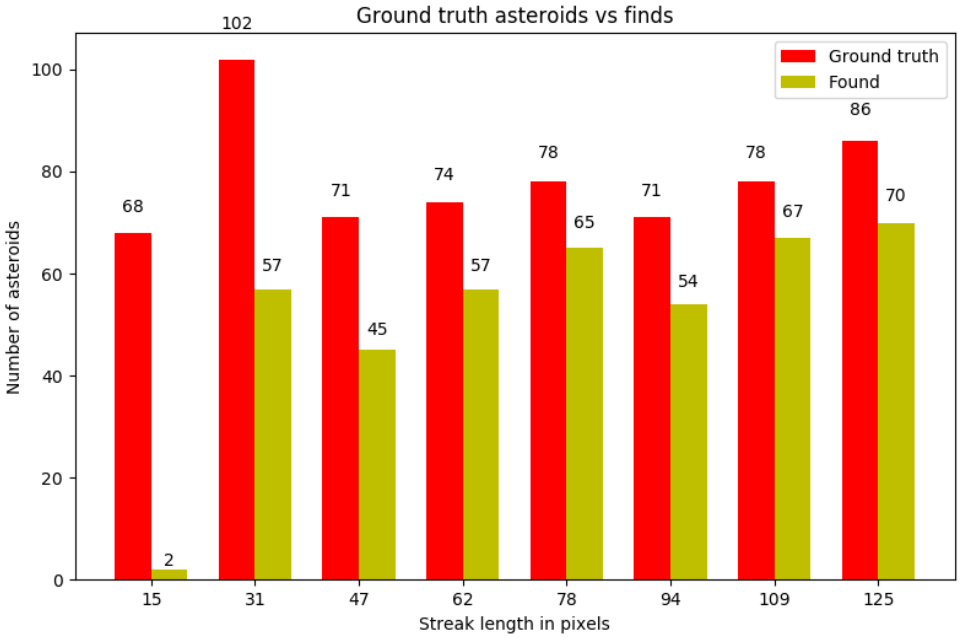

Figure A.1: Results for SSOs of different lengths after segmentation step in the magnitude range 20–21. The number of ground-truth streaks is shown as red bars, and StreakDet finds as yellow bars. The lengths go from 15 to 125 pixels.

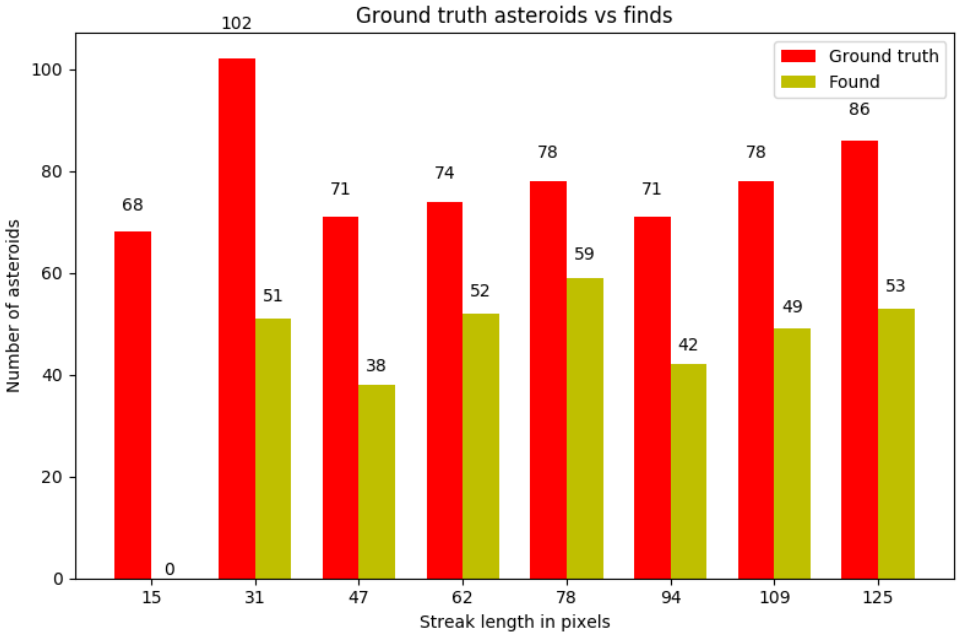

Figure A.2: Final results for SSOs of different lengths in magnitude range 20–21.

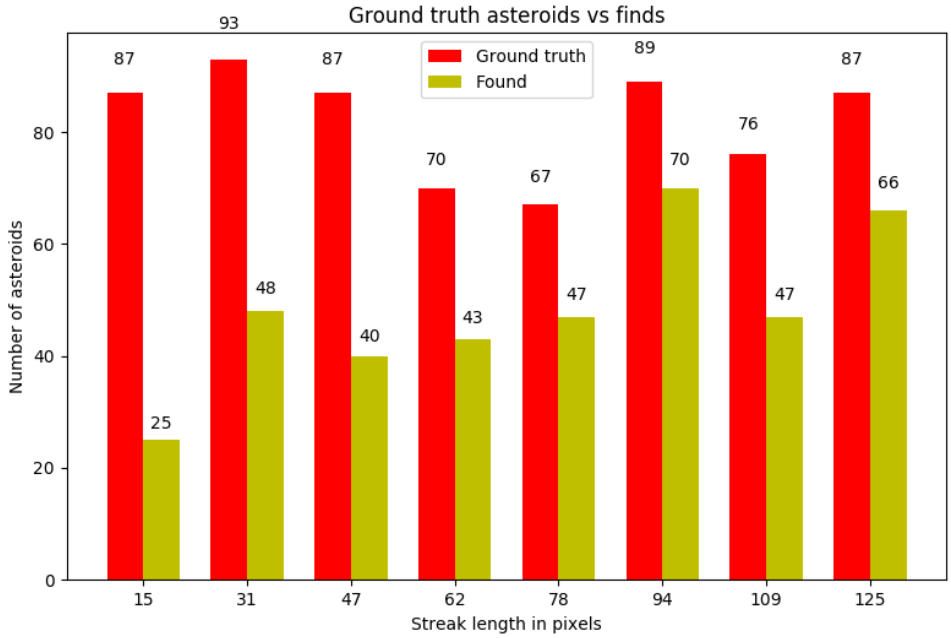

Figure A.3: Segmentation results for SSOs of different lengths in magnitude range 21–22.

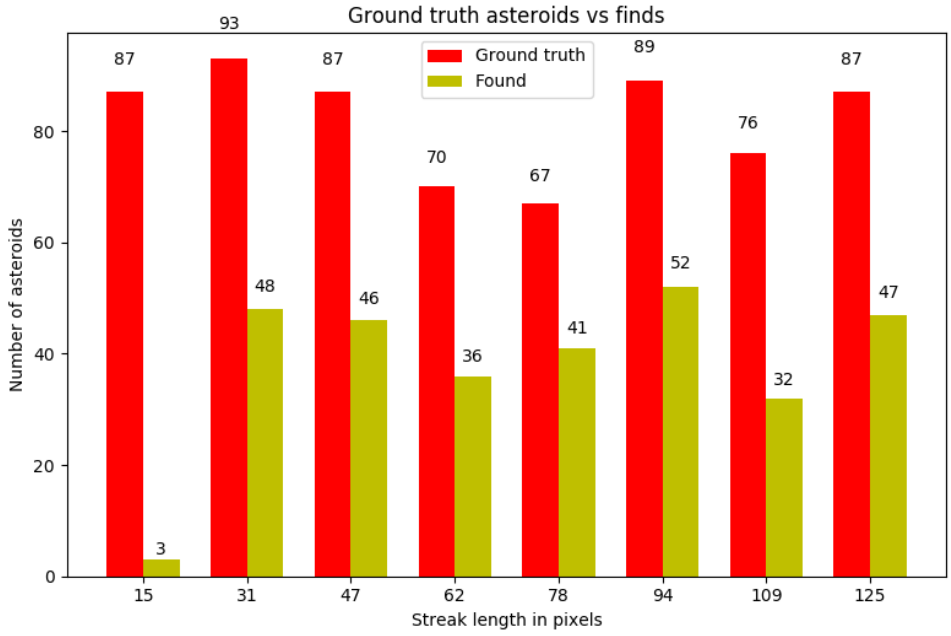

Figure A.4: Final results for SSOs of different lengths in magnitude range 21–22.

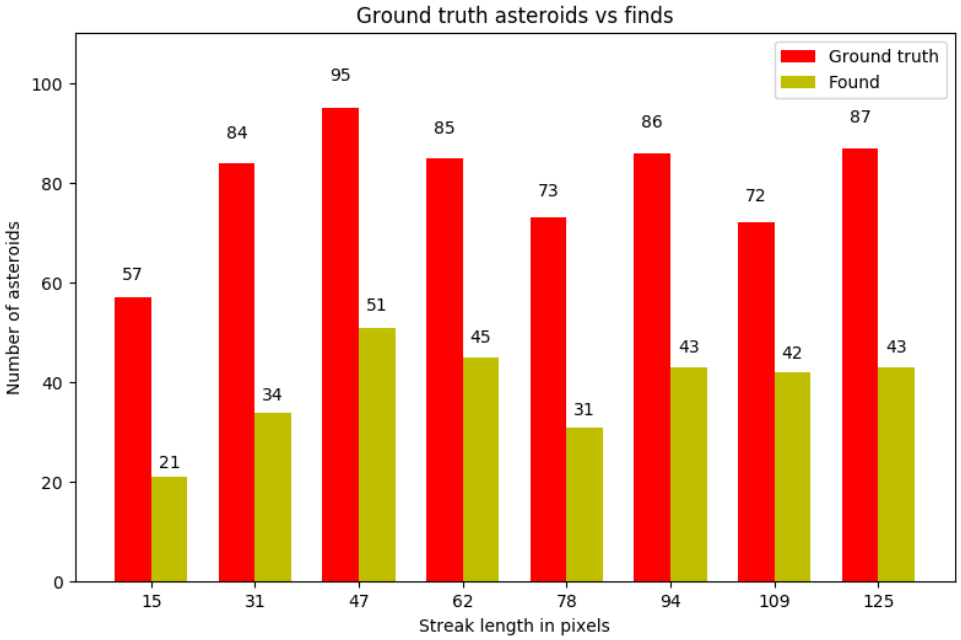

Figure A.5: Segmentation results for SSOs of different lengths in magnitude range 22–23.

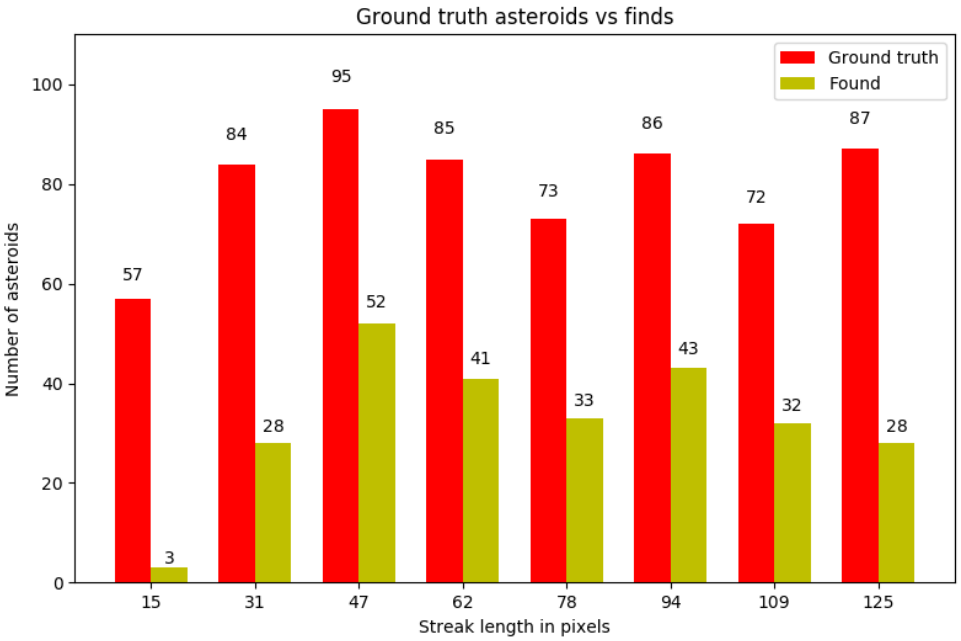

Figure A.6: Final results for SSOs of different lengths in magnitude range 22–23.

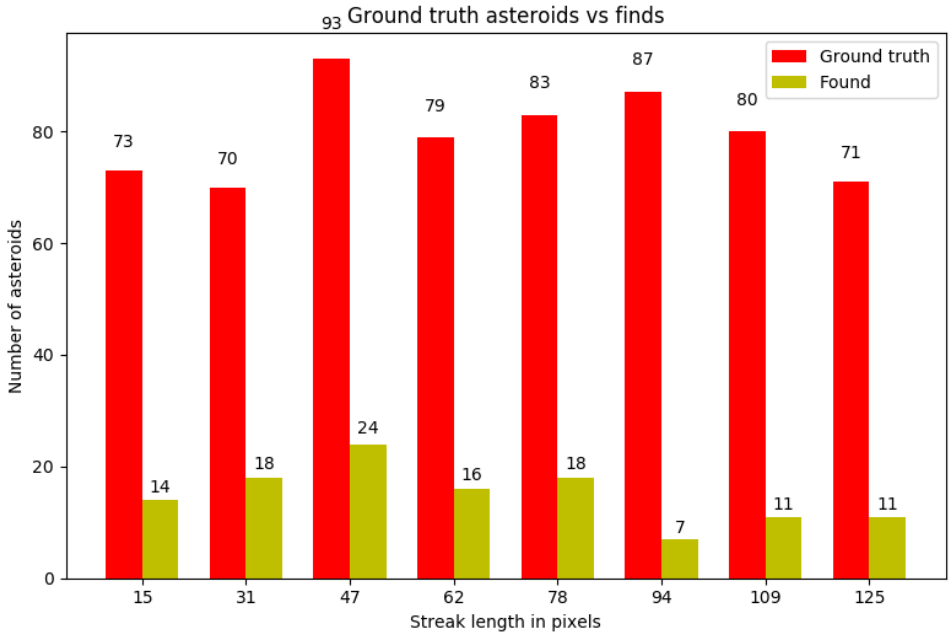

Figure A.7: Segmentation results for SSOs of different lengths in magnitude range 23–24.

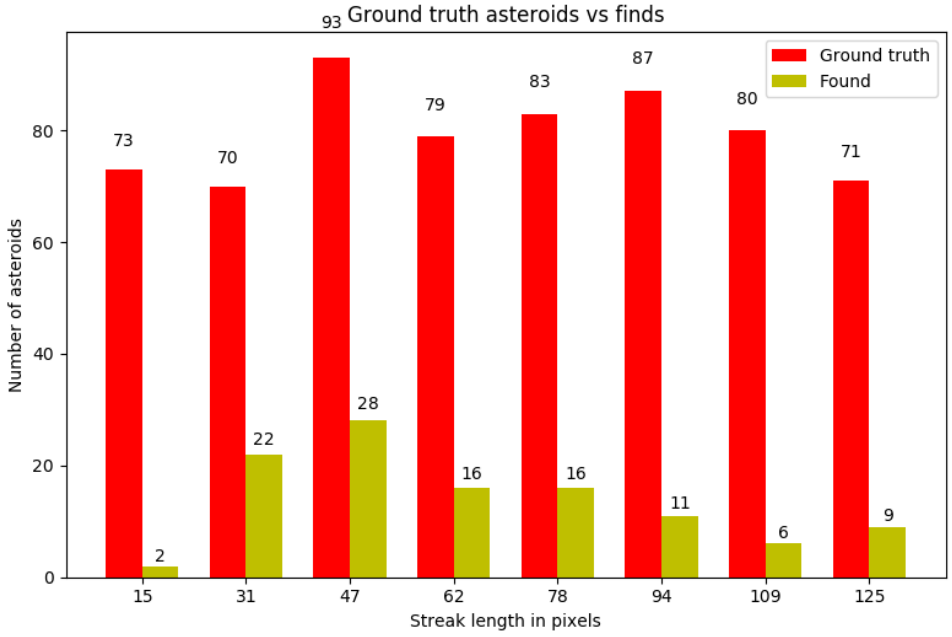

Figure A.8: Final results for SSOs of different lengths in magnitude range 23–24.

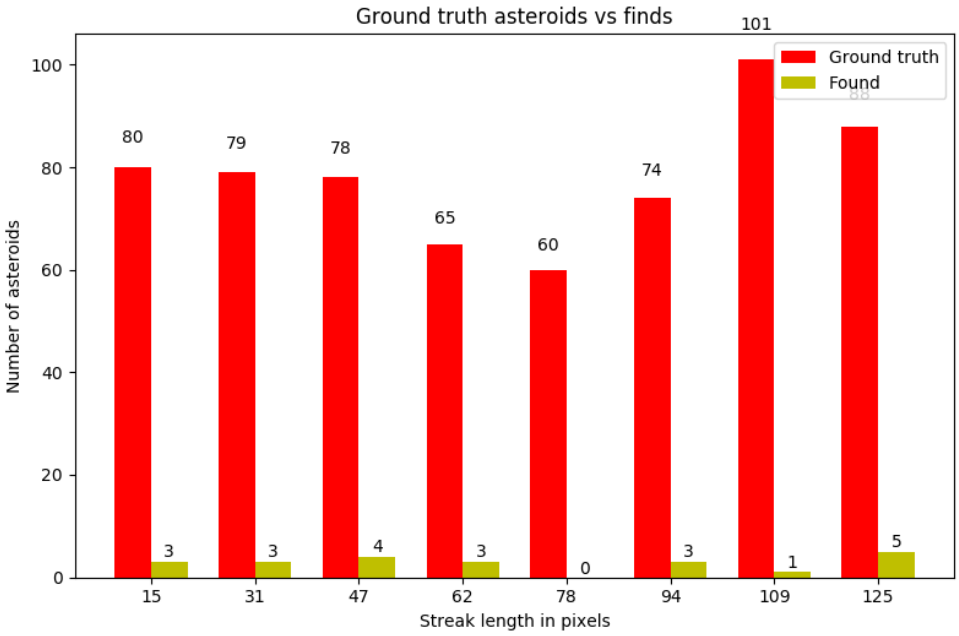

Figure A.9: Segmentation results for SSOs of different lengths in magnitude range 24–25.

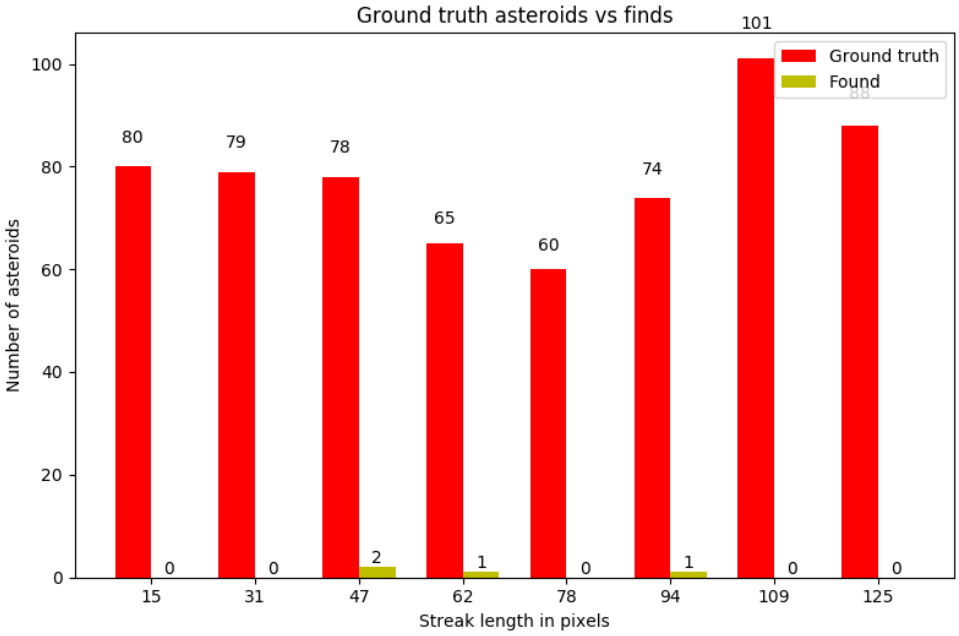

Figure A.10: Final results for SSOs of different lengths in magnitude range 24–25.

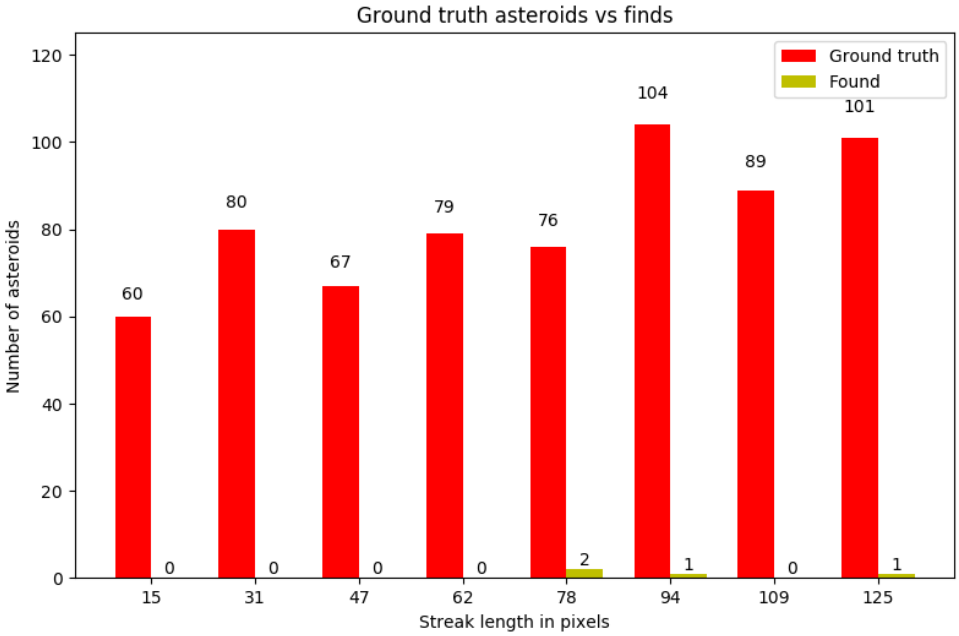

Figure A.11: Segmentation results for SSOs of different lengths in magnitude range 25–26.

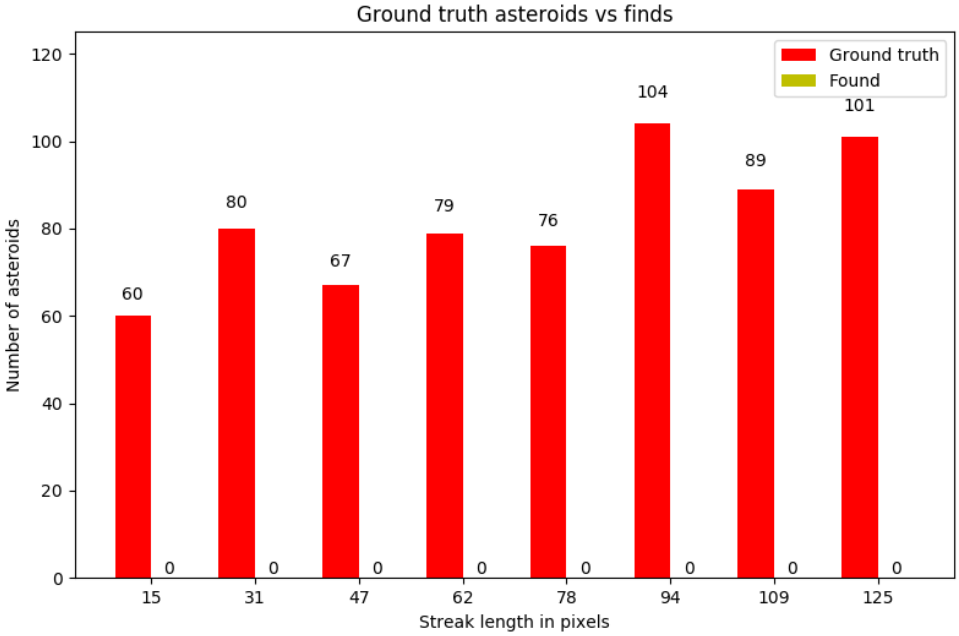

Figure A.12: Final results for SSOs of different lengths in magnitude range 25–26.

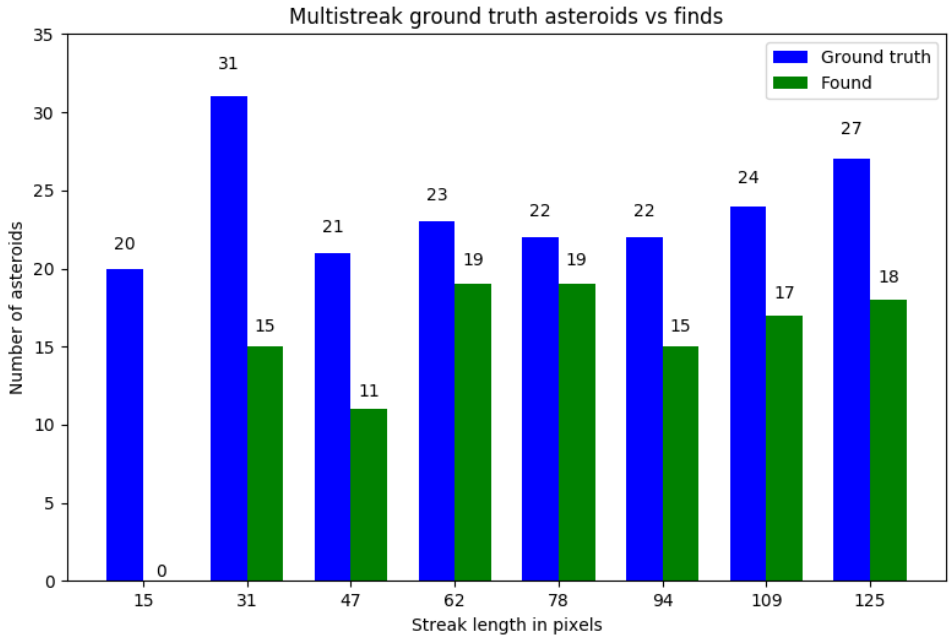

Figure A.13: Results of the multistreak analysis for magnitudes in range 20–21. The blue bars show the number of ground truth multistreaks, while the green bars show the number of StreakDet multistreaks.

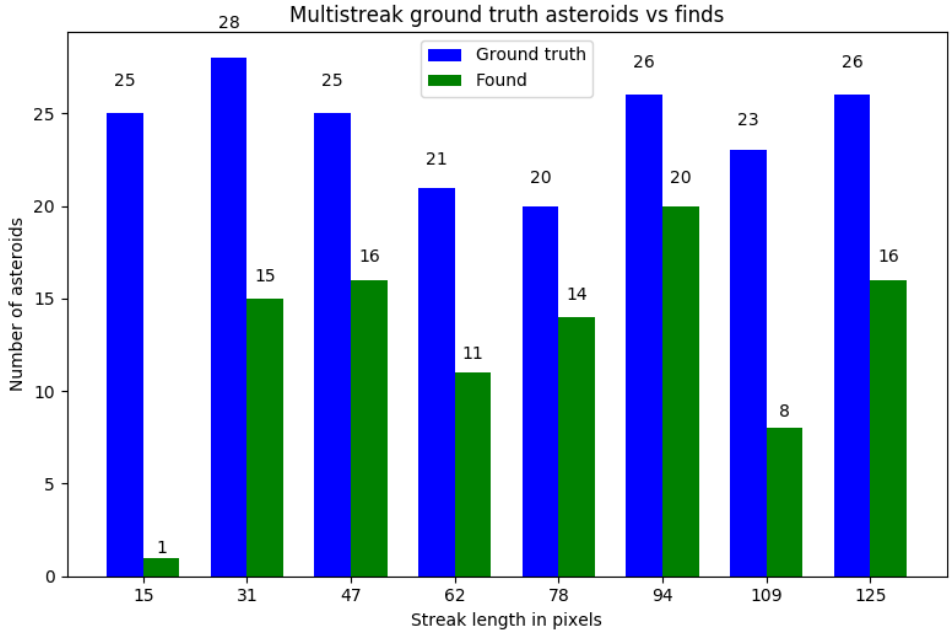

Figure A.14: Multistreak analysis results for SSOs of different lengths in magnitude range 21–22.

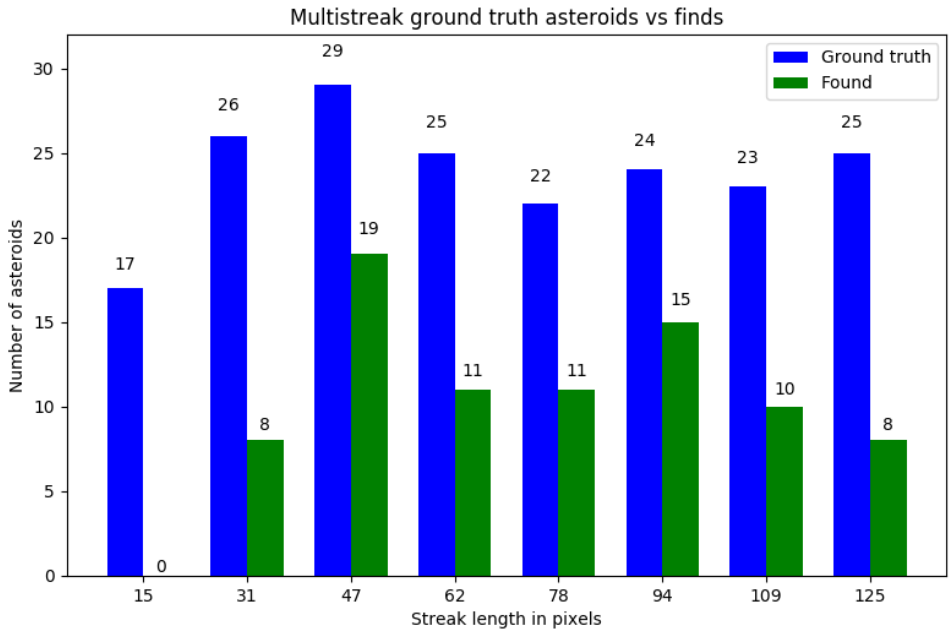

Figure A.15: Multistreak analysis results for SSOs of different lengths in magnitude range 22–23.

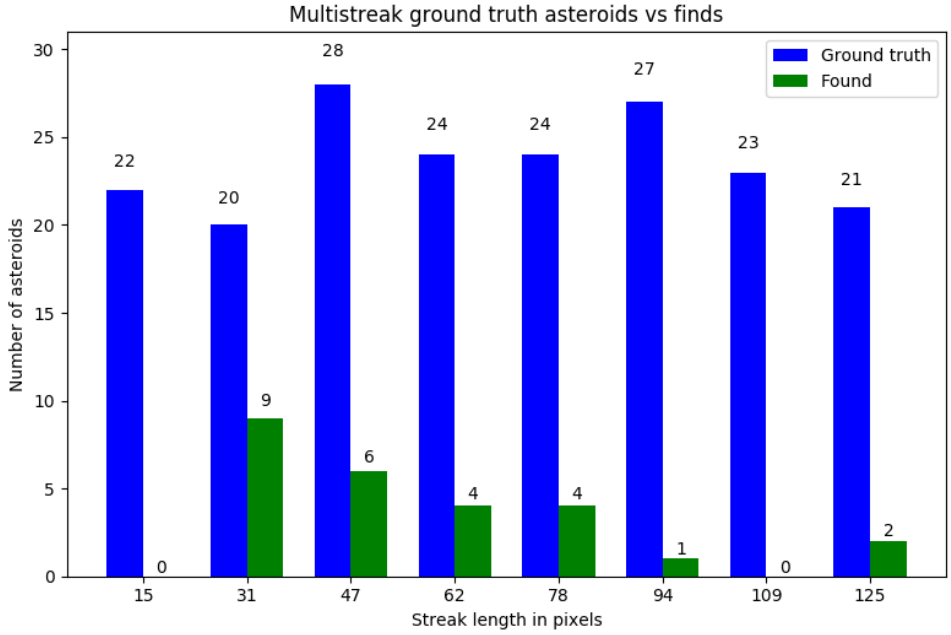

Figure A.16: Multistreak analysis results for SSOs of different lengths in magnitude range 23–24.

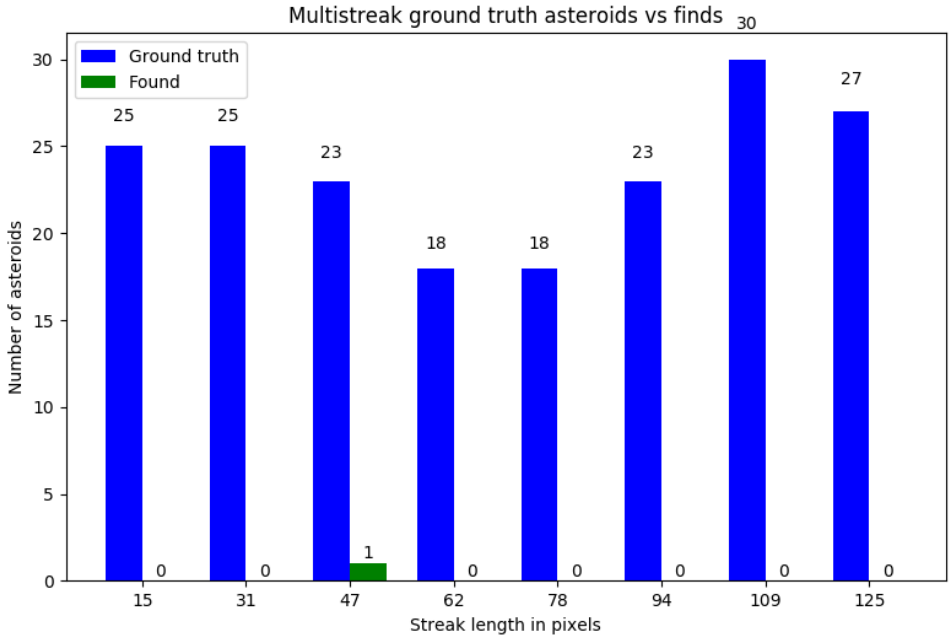

Figure A.17: Multistreak analysis results for SSOs of different lengths in magnitude range 24–25.

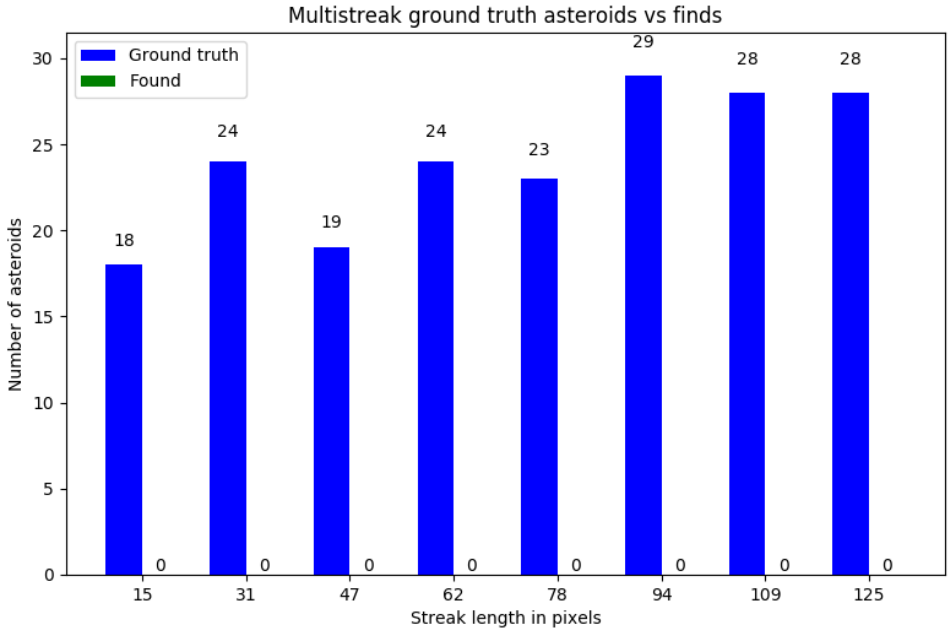

Figure A.18: Multistreak analysis results for SSOs of different lengths in magnitude range 25–26.